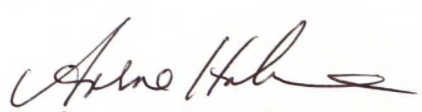
HELSINKI UNIVERSITY OF TECHNOLOGY
Department of Automation and Systems Technology

Marek Matusiak

The Design of a Mobile Ball Shaped Unit Robot for Robot Society Studies

Master's thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Technology

Espoo, 13.10.2005



Supervisor: Professor Aarne Halme

Instructor: D.Sc. Pekka Appelqvist

TEKNILLINEN KORKEAKOULU Automaatio- ja systeemitekniikan osasto	DIPLOMITYÖN TIIVISTELMÄ
Tekijä Marek Matusiak	Päiväys 13.10.2005
	Sivumäärä 64
Työn nimi Liikkuvan pallorobotin suunnittelu robottiyhteisön osaksi	
Professuuri Automaatiotekniikka	Koodi AS-84
Työn valvoja Professori Aarne Halme	
Työn ohjaaja TkT Pekka Appelqvist	

Tämä diplomityö käsittelee liikkuvan pallonmuotoisen autonomisen robotin rakennetta ja suunnittelua. Työn tarkoituksena on ollut kehittää ja rakentaa prototyyppirobotti mobiilin robottiyhteisön perusjäseneksi.

Rakennettu robotti, Minirollo, kykenee havainnoimaan ympäristöään pääasiassa kameran välityksellä sekä tarkkailemaan omaa liiketilaansa kiihtyvyysantureiden, gyroskoopin ja moottoreiden enkooderien avulla.

Suunnittelun pohjaksi on tarkasteltu ja luokiteltu aiemmin rakennettuja monirobottijärjestelmiä ja niiden jäseniä.

Minirollon päätehtävä on toimia heterogeenisen robottiyhteisön homogeenisen alaryhmän prototyypijäsenenä. Robottiyhteisön jäsenyys edellyttää robotilta kommunikointia muiden yhteisön jäsenien kanssa sekä mahdollisuutta jatkaa toimintaansa samalla kuin sen akkuja ladataan. Kommunikaatioratkaisuksi robotille on valittu Bluetooth sen robustisuuden sekä dynaamista verkottumista tukevien ominaisuuksien perusteella.

Minirollo käyttää toiminnoissaan kahta prosessoria. Mikrokontrolleri hoitaa reaaliaikaiset toiminnot, mm. liikkeenohjaukset sekä antureiden tarkkailun. Tehokkaampi sulautettu tietokone puolestaan toimii liitäntänä kameralle sekä kommunikaatiolaitteille. Lopuksi on esitetty tuloksia alustavista liike- ja paikannustesteistä.

Suunniteltu prototyyppirobotti kykenee keräämään tietoa ympäristöstään, kommunikoimaan muiden laitteiden kanssa sekä liikkumaan autonomisesti. Se luo pohjan heterogeenisen robottiyhteisön rakentamiselle ja sitä kautta avaa uusia mahdollisuuksia robottiyhteisötutkimukselle.

Avainsanat
Robotti, robottiyhteisö, kommunikointi, dynaaminen verkonmuodostus, mikrokontrolleri, pallorobotti, paikannus, odometria

HELSINKI UNIVERSITY OF TECHNOLOGY
Department of Automation and Systems Technology

ABSTRACT OF THE MASTER'S THESIS

Author
Marek Matusiak

Date
13.10.2005

Number of pages
64

Name of the thesis
The Design of a Mobile Ball Shaped Unit Robot for Robot Society Studies

Professorship
Automation Technology

Code
AS-84

Supervisor
Professor Aarne Halme

Instructor
D.Sc. Pekka Appelqvist

This master's thesis deals with the design and construction of an autonomous, mobile ball shaped robot. The aim the thesis work has been to develop and build a prototype robot to be a member of a mobile robot society.

Minirollo, the robot which has been developed, is able to detect its environment by means of a camera. It can also sense its state of motion with accelerometers, a gyroscope and motor encoders.

Various multi-robot systems which have previously been built are discussed and classified to help to set a basis for the design.

The main goal for Minirollo is to operate as a prototype member of a subgroup of a heterogeneous robot society. Membership in the robot society sets two requirements for the robot; to be able to communicate with other society members, and the capability to keep itself operational while its batteries are recharged. Bluetooth has been chosen as the communications medium for the robot because of its robustness and possibilities for dynamic networking.

Two processors are used in operating Minirollo. A microcontroller deals with the real-time functions such as motion control and operation of the sensors. A more powerful embedded computer functions as an interface for the camera and communications devices. Finally results from preliminary motion and positioning tests are described.

The designed prototype can collect information about its environment, communicate with other devices and move around autonomously. It sets a foundation for building a heterogeneous robot society and thus gives the possibility for novel robot society research in the near future.

Keywords
Robot, robot society, communication, dynamic network forming, microcontroller, ball robot, localization, odometry

ACKNOWLEDGEMENTS

The research for this master's thesis has been done in the Helsinki University of Technology's Automation Technology Laboratory as a part of the Next Generation Multi-Robot Architecture project, which has been funded by the Academy of Finland.

I would like to express my gratitude to Professor Aarne Halme for the opportunity to work at the laboratory in such an interesting project and for his guidance. The work done with the robot has taught me a great deal of the many aspects of advanced robotics, from mechanics to electronics and programming.

My instructor Pekka Appelqvist has been essential to the completion of the robot as well as my thesis, thank you for your time and support.

Furthermore, I would like to thank Kalle Rosenblad for his neverending energy to help me with the electronics, and for encouraging me when things looked grim; Tapio Leppänen for making complex mechanical parts for the robot and his constant good humor. Mika Vainio and Jari Saarinen provided me with valuable advice and ideas how to make my thesis and the robot a reality. Tomi Ylikorpi designed the complex internal mechanics thus laying the foundation for the design of the robot. Teemu Ronkka got me started with microcontrollers, and Ilya Troshin was always there to help me with the robot when I needed him. Janne Paanajärvi was the one who among a multitude of other things helped me sort out most of the bugs in my program and to bring the two processors together. Looking back, it seems that almost everyone in the Automation Technology Laboratory helped me out at some point during the project, so thank you all.

Finally I would like to thank my parents and friends for their support and Salla Mäenpää for reasons which are far too numerous to list here.

Otaniemi, 13.10.2005 Marek Matusiak

CONTENTS

1. Introduction

In this thesis an overview of the design process and construction of a mobile ball shaped unit robot is provided. This robot is being built to serve as a functioning prototype for a homogenous subgroup of a heterogeneous robot society. The processes of designing and constructing the robot are described herein so that this book can be used as a reference for designing future versions of it. The robot in question will be referred to as 'Minirollo'. The framework for the design is set by the general requirements of the project as a whole. These requirements are:

- A degree of autonomy in performing its tasks
- Interaction with the environment and the ability to communicate within the society.

After the introduction, in chapter two, the concept and the framework of the project and the robot are detailed. Also some of the ideas that were not implemented are mentioned.

In chapter three, some of the related research and development to this project are described, mostly concerning other robot societies. A method for classifying robot societies is presented.

Chapter four provides a quick overview of the Bluetooth technology and the possibilities it offers for the dynamic network management. These are essential requirements in the project. This was because Bluetooth is the primary communications protocol for the robot society and because of the dynamism of the communication network formed by multiple mobile robots.

In chapter five, the mechanics, electronics and software designs are presented and discussed. Different electronic and mechanical parts which were used in the design are presented. Their implementation process is also commented particularly if something unexpected was encountered. The software described

here deals with the lower level real-time functions, such as driving the motors and reading the sensors. The robot uses two processors, one for the lower level controls and tasks that require real-time responses, and the other for upper level communications and more processor intensive tasks. The work done has focused mostly on the lower level controller so it and the code written for it is more extensively detailed.

In chapter six the results from testing the robot are given. These tests deal with localization by using odometry.

The main part of the thesis is concluded with chapter seven which attempts to give some insight into possible future development and applications involving Minirollo.

Two appendices show details of the electronics design and the communications protocol and the command interface used in communicating between the two processors.

The group involved in the project varied in size during the project but at the time of writing this the size of the group is ten people. The others are mostly working part time on the project. The author of this thesis was responsible for designing and constructing the robot as defined by the framework of the envisioned robot society. We have one designer who designed the more complex mechanical parts, two people focusing their attention to communications and networking and three people doing part time work on another robot for the project (the so called Motherbot).

2. Concept of the System

2.1 Overview

The purpose behind the design and construction of Minirollo is to build a working prototype robot for a next-generation robot society. It continues the multi-robot research tradition in the Automation Laboratory of the Helsinki University of Technology, namely the SUBMAR project, which concentrated on building and studying the behavior of a society of ball shaped underwater robots in a three dimensional underwater environment.

The concept of the system requires the building of many similar robots, which function as agents for the system. In addition to this group of homogenous robots, a single mother robot (the Motherbot) will be built. It will serve as an energy refilling station for the smaller robots and possibly transporting some or all of them. This novel concept of an autonomous robot functioning as a recharging base station for smaller robots will enable long term autonomy for the robots as well as deployment in different environments. The robots will be capable of high levels of self-organization, to form communication networks on their own and to change networks' structure dynamically. This includes forming sub-networks as well as being a part of larger networks composed of these sub-networks.

The primary purpose for building this multi-robot society is research. The research challenges lie in the dynamism of the system, in the implementation of SLAM (Simultaneous Localization and Mapping) algorithms, dealing with delayed data, and network management when some of the agents are beyond the communications range.

2.2 Control and Communications Structure

Communications form the basis for any multi-robot system. There are many types of communications as discussed in chapter four. In this robot society, which Minirollo will be a part of, the robots should be able to form a network by themselves, without the aid of an operator. This requires dynamic network forming and management. Forming and managing a dynamic network is a challenge in its own right. It is out of this thesis's scope to delve further in to this subject than is discussed in chapter four. The concept, the limitations and possibilities of dynamic networking using the Bluetooth technology, which was chosen for this project, are described in therein.

The control of the robot society can either be handled by an operator issuing commands to it or by giving more autonomy to the society to accomplish a predetermined task. In any case the Motherbot will function as leader for the smaller Minirollo robots. A single Minirollo robot could also assume leadership responsibility as well of a group of similar robots.

Communications between the possible human operator and the robots can either be done through the Motherbot or by contacting a Minirollo robot directly.

Membership in the robot society is in no way limited to the Motherbot and Minirollo. Other robots will be included in the society thus making it even more heterogeneous. This can be done just by adding them to the communications network using the same communications software, hardware and command base.

Each robot should have a level of autonomy especially when they go out of communications range. Different operating modes will be supported, as well as different mission modes.

2.3 Design of the Robot Platform

The shape of a ball is linked to the concept of recharging the robot's batteries in the mother robot. A robot thus shaped is more easily manipulated and rotated to the right orientation for recharging in the Motherbot. A sphere shaped design maximizes the internal volume with respect to the surface area, which is useful in trying to maximize the space available for actuators etc. while at the same time keeping the size of the robot small. However it poses challenges to both the design of the mechatronics of the robot and to its control as well. A second reason for the ball-shape is our laboratory's history of building ball-shaped autonomous vehicles in the past.

One of the ideas behind the mechanical design of the robot is not to necessarily spend a lot of time and in optimizing the kinematical accuracy of the mechanical structure. The errors resulting from this are to be compensated by other means such as implementing sensors which can track the motion of the robot, such as accelerometers and a gyroscope.

The monitoring of the recharging state of Minirollo will be handled partly by itself and partly by the Motherbot. Minirollo will monitor the voltages of the batteries being recharged, while the Motherbot will adjust the current flow and voltage values based on communication from Minirollo.

2.4 Application Scenario

A practical application was also proposed as a goal for the system. Called the 'Night Watchman', it involves the agents of the multi-robot system patrolling the corridors of the laboratory and asking people moving around there for identification. Identification will either be supplied to the system by means of RFID (Radio Frequency Identification), fingerprint sensors or by similar technologies. Once identified in this way, some part of person (clothing, facial features etc.) will be identified by the camera and recorded in the memory.

This visual information can then be used for the subsequent identification attempts.

3. Multi-Robot Systems

As the aim of this thesis is to provide details of the design of working prototype robot for a robot society, this chapter discusses multi-robot systems built by other robot researchers. It also tries to focus on the abilities and designs of individual robots. These individual robots are commented on some of their abilities and characteristics, which are deemed essential for working in teams. Important societal issues discussed with each team of robots feature:

- The control and communications architectures
- The requirements that being in the society in question impose on a single unit robot.

A taxonomic method for classifying (a division into ordered groups or categories) the societies and the involved robots is presented. A clear and established way of classifying different multi-robot societies helps in understanding how they work. Differences and similarities between societies will become more readily apparent. Recognizing these will be advantageous in trying to avoid design pitfalls and understanding what previous work can readily be used in the design.

In the early 1980's extensive work was done in the field of autonomous mobile robotics. The development of better algorithms and hardware led to a new field of study, that of the multiple robots working on the same task at the same time. This posed interesting new problems for researchers to tackle and it quickly became one of the focal points in mobile robotics research *(Vainio 1999)*.

In practice most multi-robot systems extend not to only robots but also to a human operator. In many research cases this human operator is left out of the definition of the system. In *(Asama et al 1989)* and more recently in *(Arkin et al 1999)* the operator is presented as an integral part of the multi-robot system. This view of a human as a part or an extension of the intelligent infrastructure

of the robots themselves has become more popular with robotics researchers in recent times.

3.1 Taxonomy for Classifying Robot Societies

A taxonomy for multi-robot systems was presented in *(Dudek et al 1993 and 2002)*. It can be used to define the general structure, limitations and capabilities of a single multi-robot system. Other taxonomies include one presented in *(Asama 1994)*, which focuses on classifying the different control architectures, instead of the broader scope provided by the method that follows.

This subchapter gives a summary of the relevant parts of *(Dudek et al 2002)* and it is used as the reference unless otherwise noted. Each characteristic of a multi-robot system resides on an axis where its values vary. These taxonomic axes and the key points in them are presented below in Table 3-1. They portray the characteristics of the system as a whole rather than those of an individual robot.

Table 3-1 Taxonomic axes for multi-robot systems. Adapted from *(Dudek et al 2002)*

Axis	Description
Collective Size	The number of autonomous agents in the collective.
Communication Range	The maximum distance between two elements of the collective such that communication is still possible.
Communication Topology	Of the robots within the communication range, those which can be communicated with.
Communication Bandwidth	How much information elements of the collective can transmit to each other.
Collective Reconfigurability	The rate at which the organization of the collective can be modified.
Processing Ability	The computational model utilized by individual elements of the collective.
Collective Composition	Whether the elements of the collective homogenous or heterogeneous.

The key points of each taxonomic axis are described below.

Collective Size

This describes the number of robots in the environment. Key points on the axis include:

SIZE-ALONE	1 robot. The minimal collective.
SIZE-PAIR	2 robot. The simplest group.
SIZE-LIM	Multiple robots. Their number is restricted by the task or the environment where they operate in.
SIZE-INF	There is effectively an infinite number of robots.

Communication Range

Most systems have limits on the direct communication range for any single robot. The range is a function of the communications medium and robot distribution. The key points on the axis follow:

COM-NONE	No direct communication is possible between the robots. However, indirect communication might be possible by observing their presence, absence or behavior.
COM-NEAR	Robots can only communicate with other robots that are sufficiently nearby. This can for example be because of limits imposed by signal strength or physical barriers and the like. Communications can be considered to be COM-NEAR if the communication range is smaller than the maximum separation of the robots during the execution of the task at hand.
COM-INF	Robots can communicate with any other robot. Communications can be considered to COM-INF if the maximum separation of the robots in process of executing a task is smaller than the maximum communications distance.

Communication Topology

It might not be possible for a robot to communicate with some of the robots even if requirements for communication range are met. A particular hierarchy might have to be followed in communications. This hierarchy is described by the following points:

TOP-BROAD	Messages can be broadcasted allowing every robot to communicate with all of the other robots. Selective sending of messages to a certain element of the group is not possible.
TOP-ADD	Robots can communicate with each other based on a name or address.
TOP-TREE	Robots are linked in a tree like structure and can only communicate through this hierarchy. Removal of key nodes will result in a breakdown of communications.
TOP-GRAPH	Robots are linked in a general graph. Communications must pass through particular nodes to reach nodes further away. It is more redundant than a tree like structure and removing some of the nodes might not break it down.

Communication Bandwidth

Communication maybe inexpensive if the robot for example has a specific channel for communication. Or it may be expensive if the robot is prevented from doing other things while communicating. The cost of communication is compared here to the cost of moving the robot itself. Sample points on this axis include:

BAND-INF	Communication is free. Communications cost and overhead are so small that they can be ignored. This is a common assumption in theoretical robotics models.
BAND-MOTION	Communication costs approximately as much as moving between locations. From (Dudek et al 1993): "This can be thought to be similar to the mechanism by which bees communicate with each other by performing an intricate dance that is observed by other bees in the neighborhood."
BAND-LOW	Communication has a very high cost compared to movement suggesting a high degree of independence for the robots.
BAND-ZERO	No communication is possible and the robots are unable to sense each other. This makes coordinated behavior nearly impossible.

Collective Reconfigurability

This signifies the rate of spatial reorganization of the robot collective and the dynamism of their communications networks. This depends on the ability of the robots to move about, their speeds and possible routes that they can take as well as on the limitations of communications range, topology and bandwidth.

ARR-STATIC	The robots have static topology by which they must use.
ARR-COMM	Robots that can communicate can re-arrange themselves according to a specified topology or topologies.
ARR-DYN	The relations between robots are dynamic and can change arbitrarily.

Processing Ability

This axis deals with the processing ability of each collective unit. The endpoint for this axis is Turing machine equivalency. It does not have a set start point or a zero point. Even if an individual robot has limited processing capacity, the collective as a whole might be able to do complex calculations by distributing parts of the computation between its members. *(Dudek at al 1996)* offers proof that a collective of robots operated as finite state automata is as powerful as a Turing machine which is fundamentally more powerful than a single member of the collective.

PROC-SUM	Nearly the simplest possible configuration for a robot is a non-linear summation unit, like a neural network. This might be a much too simple configuration for a robot.
PROC-FSA	A finite state automaton. Certain stimuli lead to certain limited number of responses dependent on internal state.
PROC-PDA	Push-down automaton. A finite state automaton with a stack or in other words memory which influences decisions taken by the robot.
PROC-TME	Turing machine equivalency. This is the computational model assumed by most robotics systems.

Collective Composition

Even a physically homogeneous group of robots might behave differently thus being heterogeneous from a programming standpoint. The sample points on this axis aim to characterize this.

CMP-IDENT	The team of robots is uniform in both form and function, while allowing them to take different roles based on environmental or stochastic factors.
CMP-HOM	A homogeneous collective is made up from robots that are physically similar.
CMP-HET	A heterogeneous robot society is not physically uniform and they also generally behave differently as well.

In *(Balch 2002)* the degree of homogeneity is treated as an emergent property instead as a predefined one. It correlates with the performance of the collective in performing a certain task. A model for calculating that performance is also detailed.

3.2 Related Societies Labeled Taxonomically and Described

In *(Dudek et al 1996)* a full taxonomic labeling of some of the more prominent multi-robot teams up to the year 1996 as well as some examples outside of robotics study has been made. Table 3-2 below is adapted from *(Dudek et al 1996)*.

Table 3-2 Sample collectives taxonomically labeled. Adapted from *(Dudek et al 1996)*

Collective	Year	Size	Comm. range	Comm. topology	Comm. bandwidth	Reconfigur-ability	Unit processing	Composition
automobiles		SIZE-LIM	COM-NEAR	TOP-BROAD	BAND-MOTION	ARR-DYN	PROC-TME	CMP-HET
bees		SIZE-INF	COM-NEAR	TOP-BROAD	BAND-MOTION	ARR-DYN	PROC-TME	CMP-HET
combat aircraft		SIZE-LIM	COM-LONG	TOP-BROAD	BAND-INF	ARR-DYN	PROC-TME	CMP-HET
wolf pack		SIZE-LIM	COM-NEAR	TOP-BROAD	BAND-MOTION	ARR-DYN	PROC-TME	≈CMP-HOM
Aguilar	1995	SIZE-LIM	COM-NEAR	TOP-GRAPH	BAND-INF	ARR-COMM	PROC-TME	CMP-HET
Anderson	1995	SIZE-LIM	COM-NEAR	TOP-ADD	BAND-INF	ARR-STATIC	PROC-TME	CMP-HET
Arkin	*1993*	*SIZE-LIM*	*COM-NEAR*	*TOP-ADD*	*BAND-INF*	*ARR-STATIC*	*PROC-TME*	*CMP-HOM*
Brown	1995	SIZE-PAIR	COM-NEAR	TOP-BROAD	BAND-MOTION	ARR-STATIC	PROC-TME	CMP-HET
Causse	1995	SIZE-LIM	COM-INF	TOP-TREE	BAND-INF	ARR-STATIC	PROC-TME	CMP-HET
Dickson	1995	SIZE-LIM	COM-NEAR	TOP-BROAD	BAND-LOW	ARR-DYN	PROC-TME	CMP-HOM
Grabowski	*1999*	*SIZE-LIM*	*COM-NEAR*	*TOP-GRAPH*	*BAND-INF*	*ARR-DYN*	*PROC-TME*	*CMP-HET*
Habib	1992	SIZE-LIM	COM-INF	TOP-ADD	BAND-INF	ARR-DYN	PROC-TME	CMP-HOM
Hackwood	1992	SIZE-LIM	COM-NEAR	TOP-GRAPH	BAND-INF	ARR-DYN	PROC-TME	CMP-HET
Kurabayashi	1995	SIZE-LIM	COM-NEAR	TOP-ADD	BAND-INF	ARR-STATIC	PROC-TME	CMP-HOM
Kurazume	1995	SIZE-LIM	COM-INF	TOP-ADD	BAND-INF	ARR-STATIC	PROC-TME	CMP-HOM
Marapane	1995	SIZE-LIM	COM-NEAR	TOP-BROAD	BAND-MOTION	ARR-STATIC	PROC-TME	CMP-HET
Mataric	*1992*	*SIZE-LIM*	*varies*	*varies*	*varies*	*ARR-DYN*	*PROC-TME*	*CMP-HOM*
Mataric	1995	SIZE-PAIR	COM-NEAR	TOP-ADD	BAND-INF	ARR-STATIC	PROC-TME	CMP-HOM
McLurkin	*1995*	*SIZE-LIM*	*COM-NEAR*	*varies*	*BAND-INF*	*ARR-DYN*	*PROC-TME*	*CMP-HOM*
McLurkin	*2004*	*≈SIZE-INF*	*COM-NEAR*	*TOP-GRAPH*	*BAND-INF*	*ARR-DYN*	*PROC-TME*	*CMP-HOM*
Minirollo	*2005*	*SIZE-LIM*	*Varies*	*TOP-GRAPH*	*BAND-INF*	*ARR-DYN*	*PROC-TME*	*CMP-HET*
Parker	1993	SIZE-LIM	COM-NEAR	TOP-BROAD	BAND-MOTION	ARR-DYN	PROC-TME	CMP-HOM
Parker	1995	varies	COM-NEAR	TOP-ADD	varies	ARR-STATIC	PROC-FSM	varies
Rao	1995	SIZE-LIM	COM-NEAR	TOP-ADD	BAND-INF	ARR-STATIC	PROC-TME	varies
Rus	1995	varies	varies	varies	varies	varies	PROC-TME	varies
Sandini	1993	SIZE-LIM	COM-NEAR	TOP-BROAD	BAND-INF	ARR-DYN	PROC-TME	CMP-HOM
Sandini	*1993*	*SIZE-LIM*	*COM-NEAR*	*TOP-BROAD*	*BAND-INF*	*ARR-DYN*	*PROC-TME*	*CMP-HOM*
Sekiyama	1996	SIZE-LIM	COM-INF	varies	BAND-INF	ARR-DYN	PROC-TME	CMP-HET
SUBMAR	*1995*	*SIZE-LIM*	*COM-NEAR*	*TOP-BROAD*	*BAND-INF*	*ARR-DYN*	*PROC-TME*	*CMP-HOM*
Ueyama	1992	≈SIZE-INF	COM-NEAR	TOP-TREE	BAND-INF	ARR-STATIC	PROC-TME	CMP-HOM
Yuta	1992	SIZE-LIM	COM-INF	TOP-ADD	BAND-INF	ARR-DYN	PROC-TME	CMP-HOM

A few of these collectives listed in Table 3-2 are described below in more detail. These are written in *italics* in the table. In picking which societies are presented herein the focus has been on the better known ones as well as which

are relevant to this thesis. The Robot society to which the robot described elsewhere in this thesis is referred in the table as Minirollo.

3.2.1 Task Efficiency from Communication

In *(Mataric 1992)* a robot society with twenty physically identical robots was used to perform experiments on distributed control in a distributed robot population. This was one of the first experiments with multi-robot systems that tested what effect increased communication between robot system members and good or bad sensor data had on task execution efficiency.

The amount of information that the robots could get from each other including whether they knew of each others existence was varied. It was shown that when a homing test i.e. getting from a known point to another known point was performed the more intelligence was embedded in the robots the more efficiently they performed.

Each of the robots was a 30 cm – long four-wheeled vehicle with a two pronged gripper for picking up, carrying and stacking hockey pucks. It was also equipped with a radio transmitter and receiver for communication and data collection. The robot is depicted below in Figure 3-1.

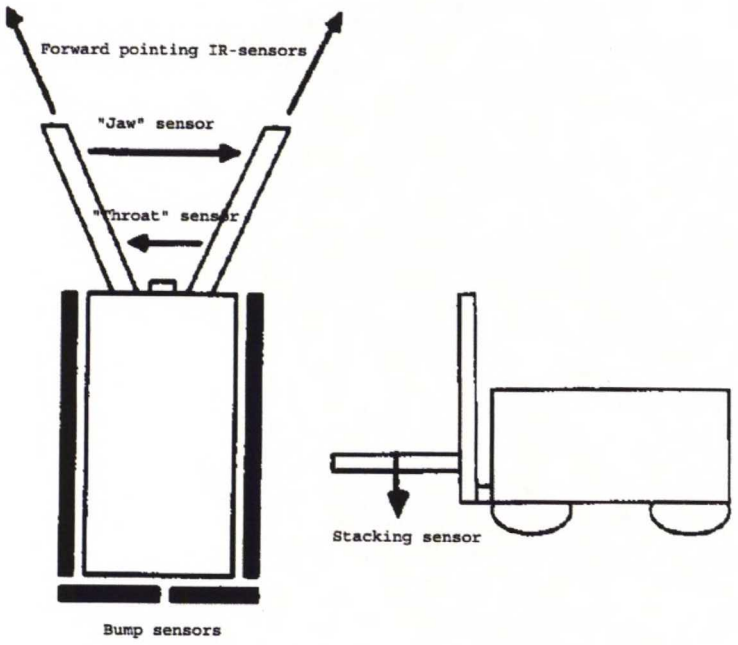


Figure 3-1 Robot used in *(Mataric 1992)*

The first test setup was with the robots unable to distinguish obstacles and other robots. The time required for the robot to find a certain destination starting from a defined point in space approached constant value. This time grew bigger steadily as more robots were introduced to the environment.

The second test was conducted with the robots being able to distinguish between obstacles and other robots. The sensing of other robots was done using the global positioning system. This gave a robot information about another robot being nearby in front of it in a forward pointing cone with a 30 cm radius before its proximity sensors could detect it. This in turn triggered a behavior called "social avoidance" which usually resulted in the robots steering clear of each other. If for some reason two or more robots managed to get in each others way, the basic obstacle avoidance forced them to back up and not collide. The results with this experimental setup demonstrated that this simple protocol was sufficiently powerful to keep robot interference from eliminating collective benefits.

The third test case expanded on the second one by adding the capability to the robots to detect each other within a 90 cm radius instead of only a forward

pointing cone. Each robot thus got a measure of population density and the population gradient. Simple attractive and repulsive behaviors were used for homing and avoidance respectively. The addition of heading information from other robots allowed coordinated movements as a team, or flocking, if several robots were heading in the same direction. This helps groups of robots to reach the same goal without great interference.

The impact that communication has on reactive multiagent robotic systems is discussed in *(Balch and Arkin, 1994)*. As (*Mataric 1992)* focused on the impact that communication had on a single task, here multiple different tasks for a robot to perform were discussed, and also how communications affected their execution efficiency. The task a robotic system is to perform dictates the need for sensors and actuators, at least to some extent. However, the need for communications between robots is not so readily apparent. To test it, three different task scenarios were devised. These were the *Foraging* task, the *Consume* task and the *Graze* task.

In the *Forage* task a robot wanders in its environment until it finds an object of interest, to which it then attaches itself to and returns it to a specified home base. The *Consume* has the robot looking for an object of interest and attaching to it as above, but instead of returning it to a home base the robot performs work on it once the attachment has taken place. After *Forage* and *Consume* have been performed once, they are repeated ad infinitum. The *Graze* task differs from these that there are no specific points of interest in the environment. Instead, the object is to cover and explore it to some extent.

These tasks were both run on a simulator and tested with real robots. Three Denning mobile robots were used. Each of them has three-wheeled kinematically holonomic suspensions and a ring of 24 ultrasonic range sensors. The robot is shown below in Figure 3-2.

Figure 3-2 Stimpy, a Denning mobile robot *(Georgia Institute of Technology)*

Communications between the robots were simple. They could observe each others internal states or in other words the task they were currently carrying out. If a robot was grazing, its mission would be to look for useful work to be done. If the robot was on the *Forage* or *Consume* task it had already found useful work to do and it would be useful for other robots to join it in its task. In this case a robot executing either of these tasks would send a broadcast message to other robots about what its current goal was.

Simulation results show that the *Forage* and *Consume* tasks clearly benefited from added communications. However, the *Graze* task did not really benefit from communications at all as other units in the simulation left a physical trail of evidence about their movement around in the environment. This was because the places they visit were physically modified, which could be observed by other robots. These results were backed up by tests with robots with the performance in the tests being only somewhat degraded in comparison to simulation results.

3.2.2 Robotic Swarms

Two large collectives of miniature robots are described in *(McLurkin 1995)* and *(McLurkin 2004)*. They both feature homogenic swarms of behavior-based robots, with a miniature micromechanical design and with the integration of multiple actuators and sensors to a very small robot. The really interesting and groundbreaking thing about these collectives is that the latter one features about 200 robots. It shows that with the right programming and hardware relatively simple commands issued to single robots can be used to make complex group level behaviors. The first one, *(McLurkin 1995)*, is presented more as a reference to the second one. Both of them have been classified in Table 3-2.

In *(McLurkin 1995)* the robotic society was meant to resemble an ant colony. It used a subsumption based behavioral control scheme. The robots had different societal tasks they could perform, for example playing tag or capture the flag.

The sensor setup for a robot can be seen in Figure 3-3.

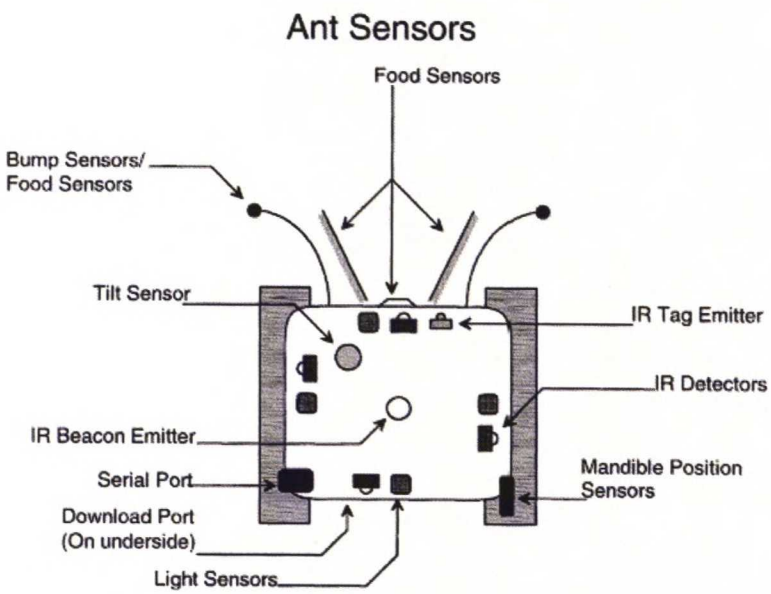


Figure 3-3 Ant sensor setup *(McLurkin 1995)*

An 8-bit Motorola 68HC11E9 microprocessor running at 2MHz was used in controlling each robot. The IR Beacon Emitter was used for communication with other robots up to a distance of six inches (15 cm). It could transmit different pulse sequences to make various signals. Robots coming in to close proximity (2.5 cm) with another could be tagged using the IR tag emitter. The IR Detectors functioned as the receivers for the signals. A total of six robots were used in the experiments.

In *(McLurkin 2004)* the case of six relatively simple robots was rather dramatically expanded to include about 200 much more complex ones. Commercially available robots, SwarmBots from iRobot, were used in constructing this multi-robot system. The SwarmBot and its technical details can be seen in Figure 3-4 below.

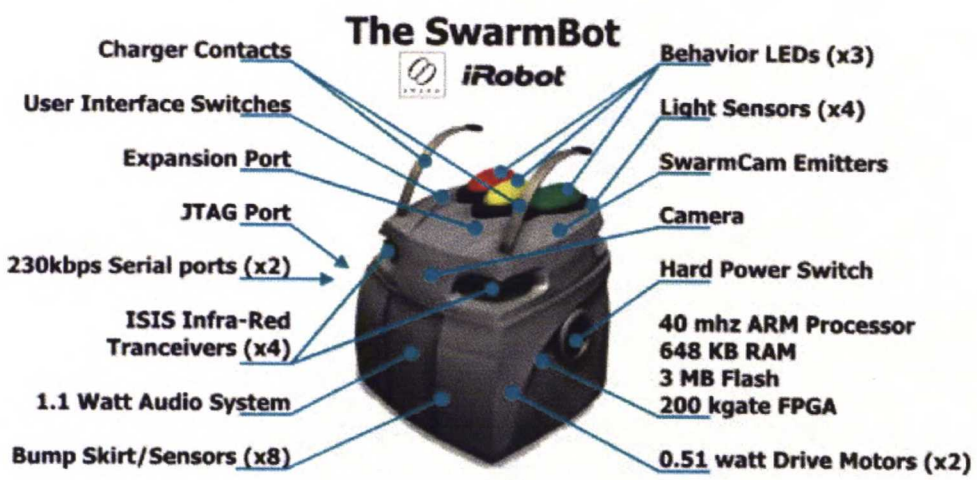


Figure 3-4 The iRobot Swarmbot *(McLurkin 2004)*

Communication was handled with four iRobot ISIS infra-red transceivers. Each robot had a unique address which enabled sending messages between two separate units. Messages advanced thru the swarm of robots in a wave like manner, each robot forwarding the message to another robot until the recipient is found or until the message had hopped through to maximum number of robots. In addition to communicating with other robots, the transceiver modules allowed each robot to determine the range, bearing and orientation of another robot within a range 50 cm with an accuracy of two degrees and two centimeters. A 32-bit microprocessor handled the control of the robot.

The robots had been programmed to have different behavioral levels, ranging from basic robot motion thru primitive commands to more complex pair and group level behaviors. The group level behaviors included avoidance of other robots, dispersing away from a single other point in space (another robot), to disperse uniformly in some space, following the leader, orbiting a group, navigating the network of robots to a single other robot, to swarm around and on a single point in space, to cluster in to groups and separate the groups form each other and to find the robots which are on the edges of the network.

3.2.3 Heterogeneous Multi-Robot Teams

The smallest unit in a hierarchical heterogeneous multi-robot team consisting of robots of different sizes and functions was described in *(Grabowski et al 1999)* and *(Navarro-Serment et al 2002).* Called the *Millibot*, this robot of miniature dimensions (7 x 7 x 7 cm) was a modular sensor platform used by larger operators to traverse and scout places only small robots can access. This has some similarities with the robot society that it used as a framework for building the robot described in this thesis. Also in our case a larger robot (the Motherbot) will function as a leader for a smaller group of robots.

The Millibot had a mobile platform which housed a main processor and allowed optional sensor and communications modules to be connected to it via an I^2C – bus. Typical sensor modules for a Millibot included sonars, infrared proximity sensors and a camera. Communication between units was handled by a radio frequency (RF) transmitter and receiver. Figure 3-5 shows the members of the multi-robot system.

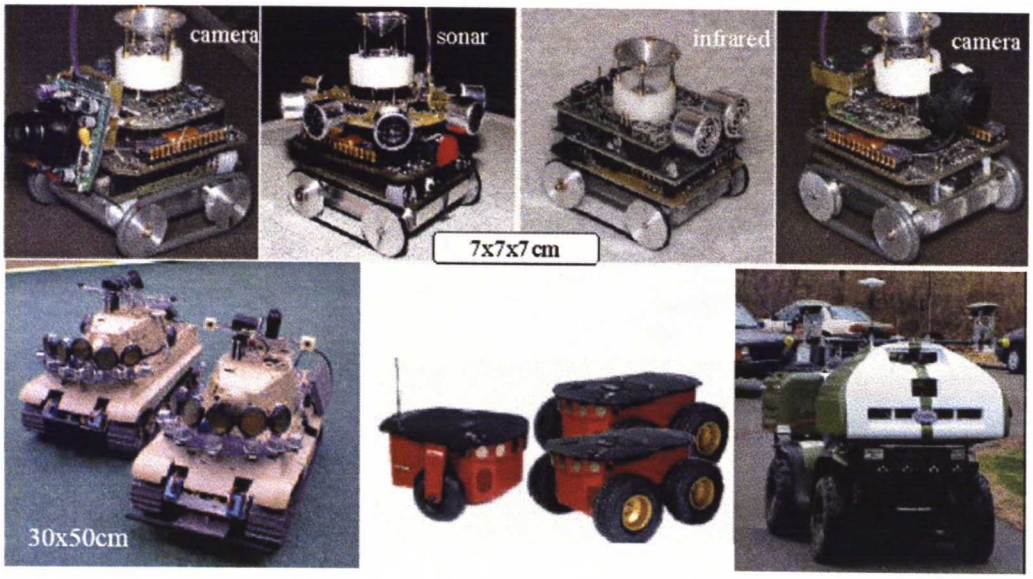


Figure 3-5 Hierarchical team of robots ***(Grabowski et al 1999)***

The robots used collaborative localization in addition to their own positional data. It was handled by a combination of sending a pulse from the sonar at the speed of sound and a pulse from the RF transmitter at the speed of light from one robot to another. The arrival time between the pulses is calculated and thus by three robots could figure out their spatial relations by using trilateration. Other members of the team included medium sized tank and Pioneer robots and large sized All Terrain Vehicles (ATVs). A larger robot (an ATV, a tank or a Pioneer robot) with more computational power functioned as a team leader for a team of smaller robots. It pooled up and merged the information gathered by the smaller units to form a larger map from the environment explored.

3.2.4 Preceding Work

A study of an underwater robot society, SUBMAR, was presented in *(Vainio 1999)* and *(Appelqvist 2000)*. It was the first one actually constructed with real robots at the Automation Laboratory of the Helsinki University of Technology. The SUBMAR robot society can be seen as a precedent to the society of which Minirollo will be a part of. Itself it was preceded by a simulation study which is described in *(Halme et al 1993)*.

The SUBMAR society consisted of homogenous robots which could maneuver up and down in water and orient itself by using two water tanks equipped with actuators. Their projected operating and final test environment was inside water pipes or tubes and their mission objective was to find and destroy algae growths by poisoning them. The chemical used in poisoning the algae, $KNnO_4$, was housed in a carrier tank from where it was released when the presence of algae was detected. These robots were ball-shaped and used the Infineon C166 16-bit processor as their main controller, which is incidentally the same processor as the one used in the Minirollo robot. Communications in the SUBMAR society are handled by radio frequency modems.

A unit member of the SUBMAR robot society is depicted below in Figure 3-6.

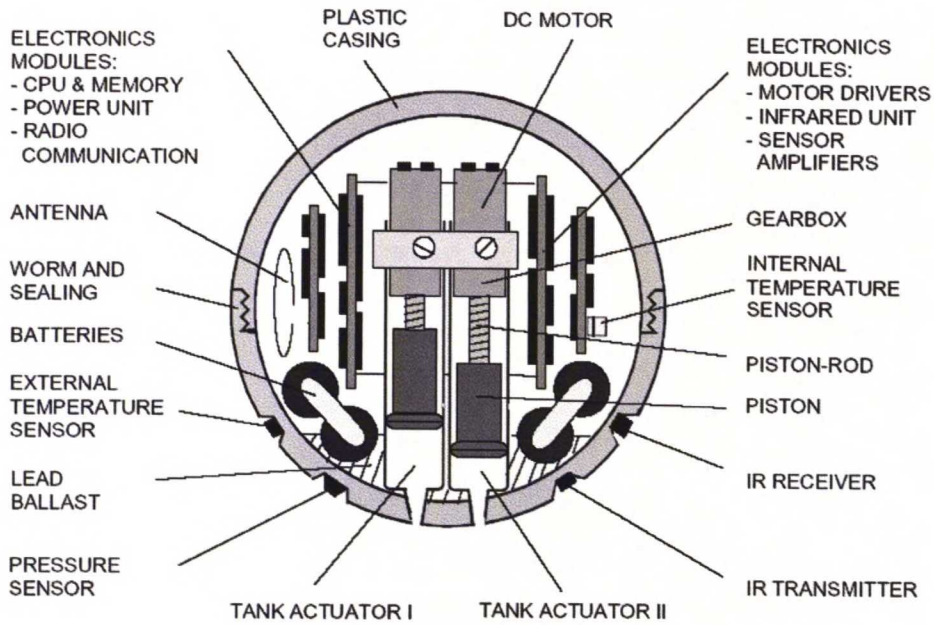


Figure 3-6 Cross-Section of a SUBMAR Proto III Robot ***(Appelqvist 2000)***

Experiments run in the test environment indicate an increase in the performance of the society when communicative interactions increased. However this result could only be achieved if the sensory data collected by the sensors was good enough.

3.3 Summary

Looking at the different multi-robot systems constructed in the past, it seems that the most important abilities a member of a society can have are:

- To observe its environment
- To interact with the environment in executing a certain task
- To communicate with other society members
- To divide a high level task command into low level commands and to execute them.

Partly based on the review of previous work described in chapters 3.1 and 3.2 and the following design decisions have been taken and implemented in the

prototype. For observing the environment and making decisions based on sensor data, a computer with high processing power has been implemented. It is described in chapter 5.2.1 Processors. Communication is rather useless unless there is relevant data to share between the communiqués. By using the camera and other sensors, data can be gathered which can be advantageous to the society in executing a task. A communications protocol between the society members has been established. It is described in Appendix B.

4. Dynamic Networking Using Bluetooth

The Bluetooth communications protocol is the choice for the communications for the following reasons:

- It is quite robust. Frequency hopping eliminates some of the bandwidth problems associated with other RF – protocols
- Dynamic network forming between mobile devices is included in the Bluetooth specification. There are some ready libraries for attempting to do this.
- With dynamic network forming and management mobile devices should be able to enter, exit and re-enter the network thus allowing communications between them.

The aim of this chapter is to provide a brief overview on Bluetooth and how it can be used by mobile devices to form a dynamic *ad-hoc* network and how the resulting network topologies will look like and function. *Ad-hoc networking* allows devices to establish communication without the constraints of a certain space or infrastructure *(Frodigh et al 2002)*. First an overview of the technologies in question will be provided, second the different topologies will be described, and third some practical implementation issues will be discussed.

4.1 Introduction

Bluetooth is a wireless communication technology that uses a frequency-hopping scheme in the unlicensed Industrial – Scientific - Medical (ISM) band at 2.4 GHz *(Frodigh et al 2002)*. With Bluetooth various mobile devices can be easily interconnected. It requires a low-cost transceiver chip to be included on each device. The 2.4 GHz frequency band is available globally with some variations in bandwidth in different countries. Each device has a unique 48-bit address defined in the IEEE 802 standard. Frequency hopping makes communication possible even in areas of electromagnetic disturbance.

Bluetooth devices come in three power classes. They are shown below in Table 4-1.

Table 4-1 Bluetooth Power Classes *(Mikéska 2004)*

Type	Maximal Output Power	Nominal Output Power	Minimal Output Power	Operating Range
Class 1	100mW (20 dBm)	-	1mW (0 dBm)	Up to 100 meters
Class 2	10mW (4dBm)	1mW (0 dBm)	0,25 mW (-6 dBm)	Up to 10 meters
Class 3	1mW (0 dBm)	-	-	0.1-10 meters

4.2 Bluetooth Specification Protocol Stack

Bluetooth handles communication using the following protocols: HCI (Host Controller Interface Specification), L2CAP (Logical Link Control and Adaptation Protocol Specification), SDP (Service Discovery Protocol Specification), RFCOMM (Serial port emulation over L2CAP), LMP (Link Manager Protocol) and BNEP (Bluetooth Network Encapsulation Protocol Specification). A brief description of each of the protocols is provided below listed from top to bottom. The reference below is *(Mc Daid)* unless otherwise noted. Figure 4-1 displays the protocol stack.

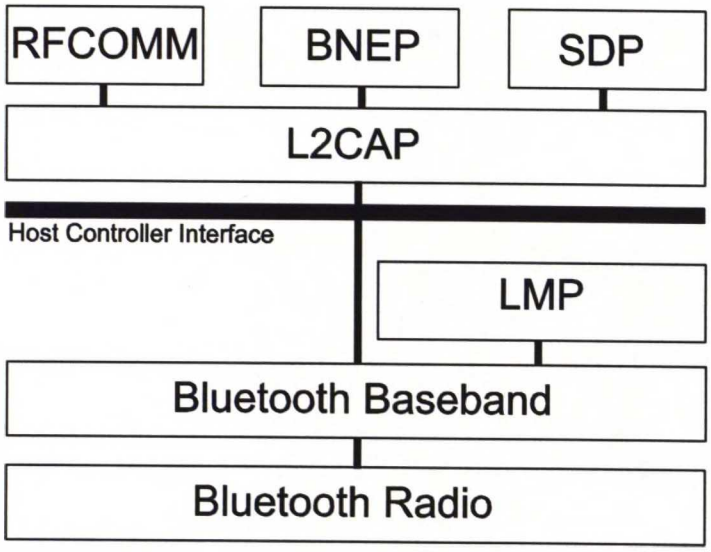


Figure 4-1 The Bluetooth Protocol Stack

SDP

The service discovery protocol provides means to discover what services are available and to determine characteristics of those services. It resides over the L2CAP layer.

RFCOMM

The RFCOMM protocol provides emulation of serial ports (RS232) over the L2CAP layer. It based on the ETSI standard TS 07.10 *(The BlueZ Project 2003)*. Using RFCOMM to communicate a single Bluetooth device can maintain connections at most to two devices.

BNEP

The Bluetooth Network Encapsulation Protocol provides an interface to IP. It is used by the Bluetooth Personal Area Network (PAN) – profile and thus provides the possibility to communicate with multiple devices. It resides over the L2CAP – layer *(Johansson et al)*.

L2CAP

The Logical Link Control and Adaptation Protocol are layered over the HCI layer. It provides connection-oriented and connectionless data services to upper layer protocols, with data multiplexing, segmentation and reassembly capabilities. Group management, i.e. the capability to map transparently groups of addresses (devices) onto piconets without needing to know the baseband routines, and Quality of Service (QoS) are also supported.

HCI

"*The Host Controller Interface (HCI) provides a standard interface to the Bluetooth baseband controller and link manager services that is independent from the host hardware implementation. This layer provides a uniform method of accessing any Bluetooth hardware. There is an addendum to the HCI specification for different host transport protocols. For each physical bus (USB, RS232, UART etc.) it defines the interface functions based on which physical bus is used, but also vendor specific implementations are possible.*" (The BlueZ Project 2003)

LMP

The Link Manager Protocol is responsible for establishing and managing physical links. The latter consists of putting slaves in particular operating states (sniff, park or hold mode) and monitoring the status of the physical channel while assuring Quality of Service *(Bruno et al 2001)*.

Baseband

The Bluetooth Baseband layer resides on top of the Bluetooth radio layer. It handles packets, links, manages synchronous (point-to-point) and asynchronous (master-to-slave broadcast) links, and does paging and inquiry to access and find Bluetooth devices in the area.

Radio

The Bluetooth radio (layer) is the lowest defined layer in the Bluetooth specification, defining the requirements for a Bluetooth transceiver device. In

the utilized 2.4GHz band 79 Radio Frequency (RF) channels are defined, spaced at 1 MHz intervals. A unique pseudo random frequency hopping spread spectrum sequence (FHSS) utilizing 32 of these channels is formed for each ad-hoc network established *(Bruno et al 2001)*.

4.3 Pico- and Scatternets

Bluetooth networks are comprised of two or more piconets. These piconets enable up to eight BT devices to connect and form a network. One device acts as a master, others as slaves to it. Up to seven devices can be actively connected to a single master. When a connection between Bluetooth devices is established, the first step is to look for other devices in range. Discoverable devices within range will respond to the device inquiry. Bluetooth addresses of the devices in question are obtained. The master decides which slave can access the channel at any given time by paging the slave in question. When a slave receives a page packet it responds to the master who then sends a Frequency Hopping Synchronization (FHS) packet to the slave. The slave acknowledges and generates a frequency hopping sequence identical to the master's. Once the paging process is completed the devices move to the connection state. The devices continue hopping using the same sequence until the connection is terminated.

Slave devices in a piconet can only communicate with the master and only immediately after being addressed by it *(Mišić et al 2003)*.

A piconet has a gross bit rate of 1Mbit/sec. If there are more than eight devices in total, some of the devices can be "parked". To communicate with a parked slave the slave has to be "unparked" thus possibly forcing the parking of another slave *(Basagni et al 2002)*.

Scatternets are networked piconets. A slave device from one piconet can act as a master in another. Additionally, a slave device can function as a bridge between two piconets, routing communications between them and thus

creating a scatternet. However, a slave can only transmit on a single piconet at any given time. It has to change its synchronization parameters before listening to different channels *(Bruno et al 2001)*. Figure 4-2 shows a sample scatternet composed of three piconets. The node which bridges piconets one and two functions as a slave in both. Piconets two and three are connected thru a node acting as a slave in net two and as a master in net three.

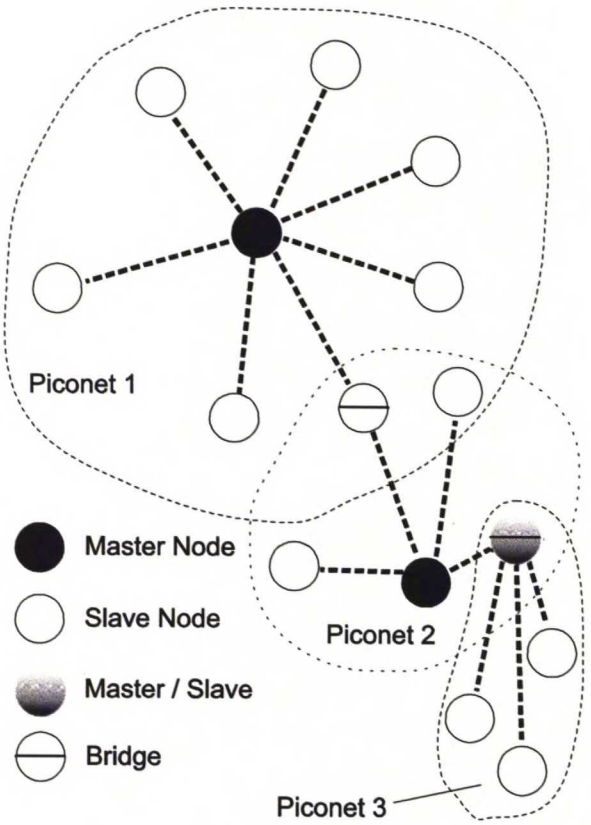


Figure 4-2 Connected piconets forming a scatternet

The number of connections that can be counted from one node to another are referred to as *hops.* This refers to the number of network nodes a packet of data has go through trying to access it's final destination. A packet traversing from a master node to a slave would go through one hop, while another packet which would go from a master to a bridge node and then to third node would go through three hops *(Melodia et al 2003)*.

The formation, structure and how a scatternet is maintained is not a part of the Bluetooth specifications. Each of these factors has a tremendous impact on the performance of the network *(Barriere et al 2003)*.

Problems in communication between piconets can arise if their number becomes too large because they have only a limited number of channels to share. Bridge nodes are critical to the flow of data so they should not have too many connections to other nodes. By definition a master node can have up to seven connections inside a piconet, thus if it also functions as a bridge communications between two nets can become bogged down *(Barriere et al 2003)*.

The routing of packets in a scatternet should work when a working net is formed, unless the network in question is quite large. In this case a problem might arise with the use of traditional proactive routing algorithms, ones that keep track of each and every node in the network. An alternative to this is to use a reactive routing algorithm, which does not try to maintain a map of the whole network, but rather routes between nodes are determined only when they are explicitly needed to route packets. Examples of reactive routing can be found in *(Macker et al 2004)* and *(Johansson et al)*.

4.4 Practical Implementation

The Linux Bluetooth stack BlueZ was the choice for this project. It offers tools for the construction of a piconet and eventually a scatternet.

An experiment with two Bluetooth devices forming a PAN was successfully conducted. However, when the devices moved out of range, renegotiation of the connection had to be done manually. With three Bluetooth devices, making one of the devices a bridge node, the formation of a scatternet should be possible. The experiment also showed that in a laboratory environment a class 2 Bluetooth device communicating with a class 1 device was able to communicate through an indoor wall at a range of ca. 15m.

The Bluetooth devices that are to be used in ad-hoc networks need to be ones that have the possibility of participating in two BT networks at the same time. This enables scatternet functionality.

Using the Linux BlueZ stack to build a large PAN one needs to make each one of the participants in the network a Linux-Bridge. However BlueZ doesn't support by default the construction of a scatternet. The topology must be defined by the user. BNEP or the PAN functions of the stack seem to reserve some of the functionality of the HCI.

According to the preliminary testing and based on the literature survey given above, Bluetooth is a suitable communications solution for the robot society. The tests confirm the ability of Bluetooth devices to build a network and maintain communications at a sufficiently long range, while the survey shows the potential it has for dynamic network management.

5. The Design of Minirollo

In this chapter the technical details of the robot are discussed. The chapter is divided in to three sub-chapters, the first for the mechanical design, the second for the electronics devices and design and the third part details the software.

5.1 Mechanical Implementation

The framework for the novel mechanical design of Minirollo is based on the concept of the robot being built inside a ball which opens into two halves. The encasing halves of the ball function as wheels. While opening the ball a tail extends from inside the back of the ball to provide a supportive force for the increased torque. The robot can be seen in Figure 5-1 below.

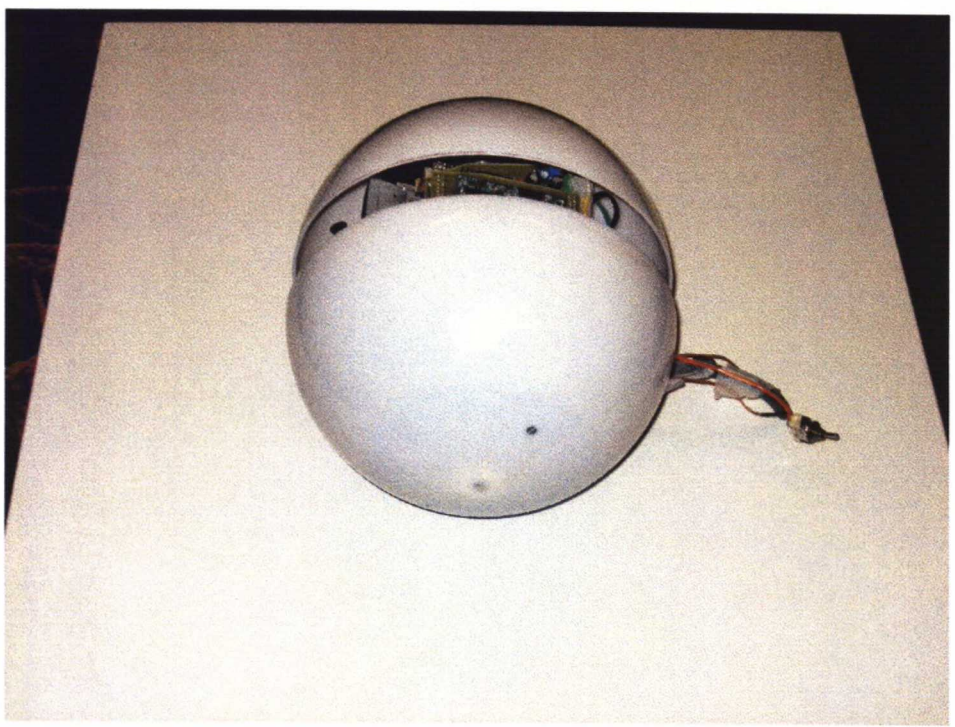


Figure 5-1 Minirollo seen from the left side

The ball is opened and closed by driving or rotating the right wheel backwards. This causes the ball either to open or close depending on the position of the opening mechanism, namely a lineguide, which more

commonly can be found in fishing equipment, more precisely in a reel. This lineguide can be seen in Figure 5-3 where it is denoted with the number 1.

Below, in Figure 5-2 the prototype can be seen from the front. Both of the processors can be seen from the picture, as well as the camera. These are described in chapter 5.2 Electronics. Below the ball shaped casing is opened up as far as it will open, for a total of 2.5 cm. The diameter of the casing is 19 cm while it is closed.

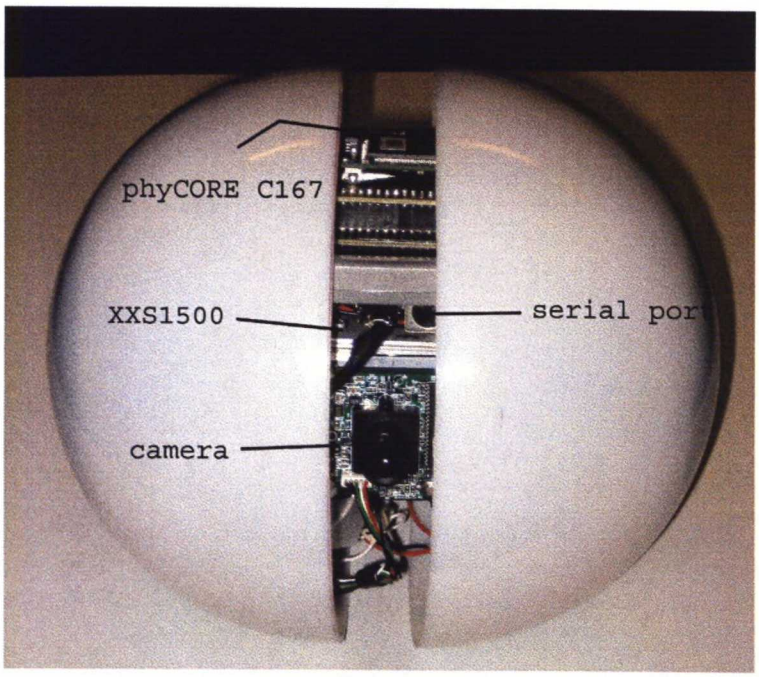


Figure 5-2 Minirollo from the front

In Figure 5-3 the interior mechanics, which are used in opening and closing the ball casing and extending the tail, are shown. The lineguide (marked 1 in the picture), moves back and forth from and to both extremes by rotating the right wheel backwards. This causes the two main pieces of the interior mechanics to either slide away from or to move closer from each other. Item marked with 2 in Figure 5-3 extends the tail when the ball is opened.

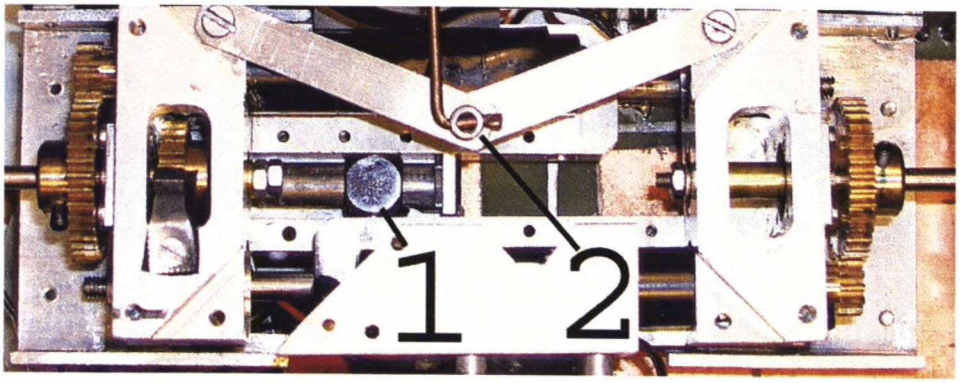


Figure 5-3 The mechanics used in opening the ball and extending the tail. Item marked 1 is the lineguide, and item 2 is part of the mechanism used in the tail extension.

The opening of the prototype is shown with four pictures in Figure 5-4. Notice the tail and the positions of the items marked 1 and 2 in Figure 5-3 and their movement in throughout the four pictures. The robot is closed in the upper left hand picture and fully open in the lower right hand one.

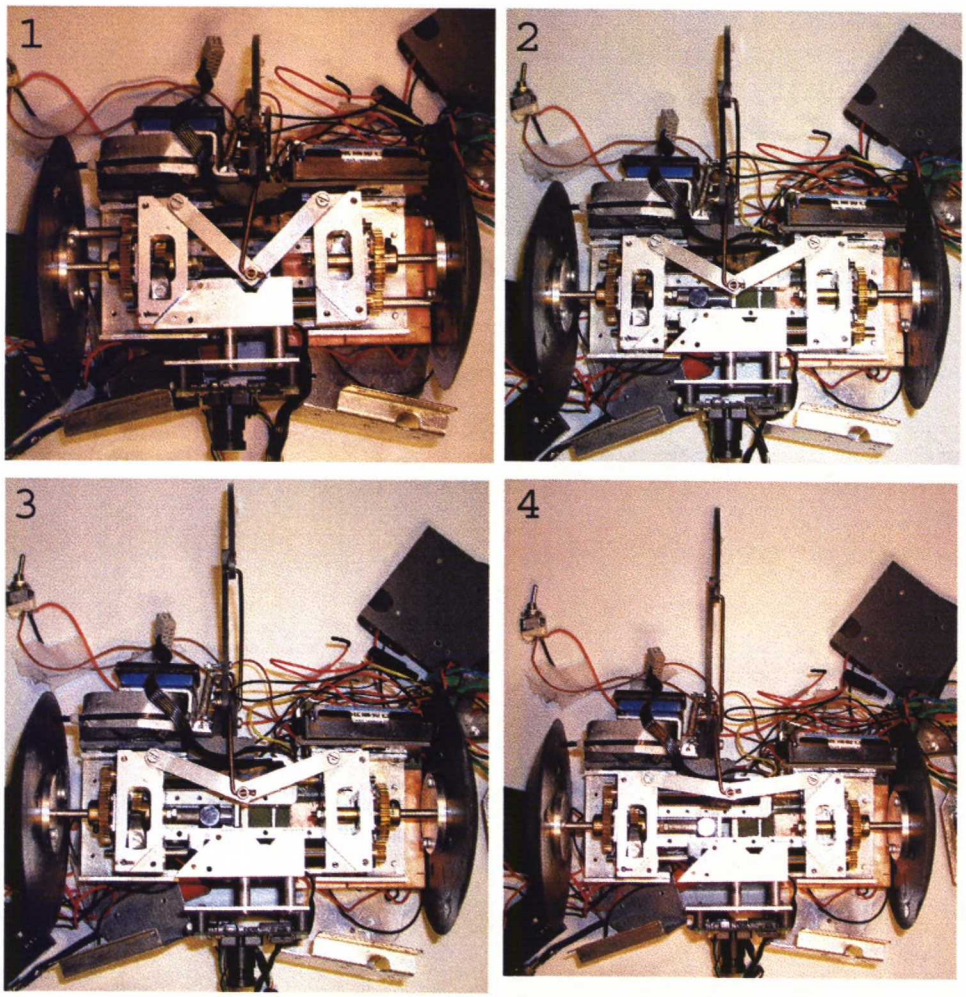


Figure 5-4 Tail extension and opening up

5.1.1 Kinematical Equations for the Design

Minirollo can be modeled as a two-wheeled differential drive vehicle. The distance between the wheels is denoted by D. When both wheels are rotated with different (and constant) velocities the trajectory which the robot traverses is shaped like a segment of the arc of a circle. The time interval for updating the positional data of the robot is small. Because of this, the effect of acceleration during this period can be neglected and the wheels can be considered to be turning at a constant angular velocity. The distance traversed by the faster moving wheel is:

$$s_1 = v_1 t_1 \tag{1}$$

and by the slower one:

$$s_2 = v_2 t_2 \tag{2}$$

If the turn radius of the slower moving wheel is r, then the turn radius of the faster moving wheel is $r + D$. The angular sector of the circular path traveled by the robot is

$$\theta = \frac{s_1}{r+D} = \frac{s_2}{r} \tag{3}$$

The turn radius of the inner wheel can be solved from equation (3) as follows

$$s_1 r = s_2 r + s_2 D$$

and

$$r = \frac{s_2 D}{s_1 - s_2}. \tag{4}$$

The change in angular orientation (change in the yaw angle) can be calculated by combining equations (3) and (4) as follows

$$\theta = \frac{s_1 - s_2}{D}. \tag{5}$$

Equation (5) is used in conjunction with the distance traveled by the wheels to keep track of the position of the robot. This is discussed in chapter 5.3.2.

5.2 Electronics

In designing Minirollo a lot of electronics design had to be done. This was in part due to the absence of a separate motor controller and the fact that two processors were used in the design. The rigorous computation needed for image processing and the network traffic handling was the reason for implementing a second more powerful processor to work alongside the 16-bit microcontroller, which in turn was used to handle the simpler and more time critical tasks.

Appendix A includes schematics and pictures of pin-layouts of the circuit boards used in the robot.

A general view of the robot as seen from the front can be found below in Figure 5-5. Some of the essential items which are visible are marked in the figure.

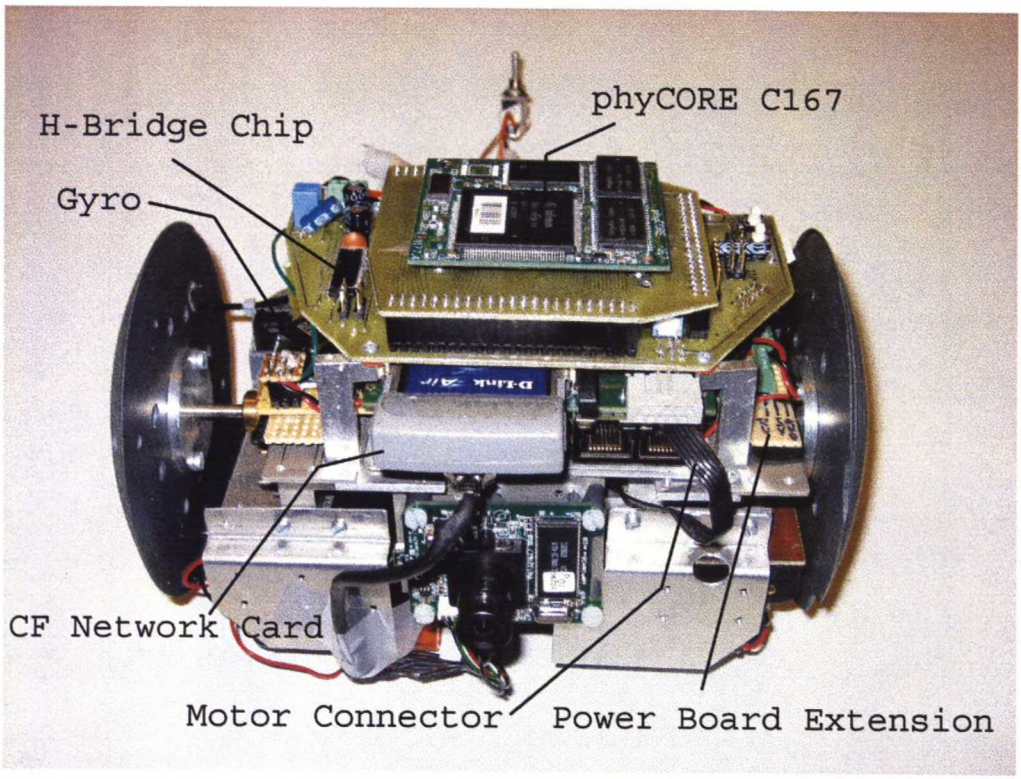


Figure 5-5 General view of the robot, casing open

Two asymmetrical circuit boards were designed for the robot. The first one serves as a mounting board for phyCORE 167. It was designed to facilitate the testing of various types of main boards. Because of the fact that microcontroller module's connectors were hard to solder by hand as the division between the legs being only 25 mils, it was thought better to solder these connectors to an interface board with a standard pin header interface (a pin spacing of 100 mils).

The second asymmetrical board designed for the robot is the underlying main circuit board. It houses the H-bridge chip which is controlled with the PWM – outputs of the microcontroller, the serial port interface, analog-to-digital channels as well as other general I/O and two accelerometers. The H – bridge chip (L293D) does tend to get very hot during operation. However this doesn't seem to affect the performance of the robot much.

Under the robot, between the mechanics and the batteries, reside two positional switches which are used to discern if the robot is closed or open.

A schematic view of the electronics of the robot can be found in Figure 5-6 below. The devices are described in more detail in the following subchapters.

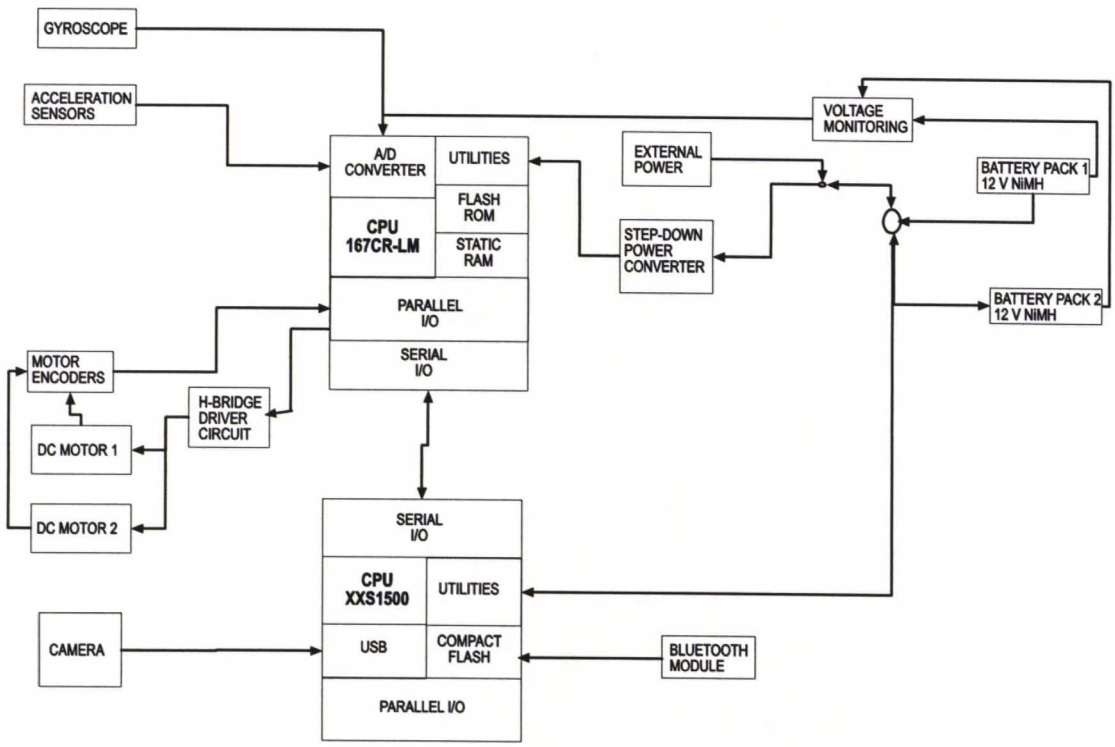


Figure 5-6 A schematic of the robot

5.2.1 Processors

There are two processors implemented in the robot, each with their distinct functions. The Phytec phyCORE 167CR microcontroller functions as the main motion and power controller and sensor interface, whereas a Linux-box, the Mycable XXS1500, is used for tasks that require more processing power. These two processors communicate with each other by a serial port connection. They are detailed in the following sub-chapters and a comparison of them can be found in
Table 5-1 below.

Table 5-1 Processor comparison

	phyCORE167	XXS1500
Available Digital I/O	48	11
Analog Inputs	16	0
Power Consumption	0.5 W	4.5 W
SRAM	256k - 1M	64M - 128M
FLASH	256k - 2M	16M
EEPROM	4k - 32k	0
PWM	yes	no
Serial Ports	two (6 pins)	two
Size (mm)	60 x 53	54 x 85.6
Year	2001	2003
Connector	0.635mm pitch (Molex)	-
Operating Voltage	5V	8-14V
Notes	5V flash voltage	USB, CompactFlash, 2 x Ethernet

The Sensor and Actuator Interface Controller

The phyCore 167 CR, manufactured by Phytec Technologie Holding AG, is a subminiature single board computer populated by Infineon's 16-bit C167CR microcontroller. It functions as the robot's sensor and actuator interface. It has a 20MHz clock cycle, 16 MB of address space and different memory sizes depending on the configuration (256kB – 1MBof RAM, 256kB – 2MB of Flash memory). It also features two RS-232 serial interfaces, on-board pulse width modulation signal generation and multi-function timers, an analog-to-digital converter with 16 multiplexed channels and a 10-bit resolution *(Phytec Meßtechnik GmbH)*. This controller was chosen for the project based on the previous experience with Phytec controller boards. Its size and the large number of available input / output lines complied well with the project requirements. The processor board is shown below in Figure 5-7, and a block diagram of the board can be seen in Figure 5-8.

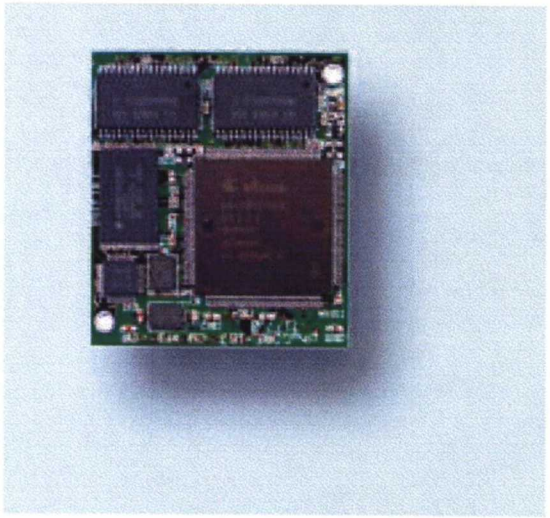
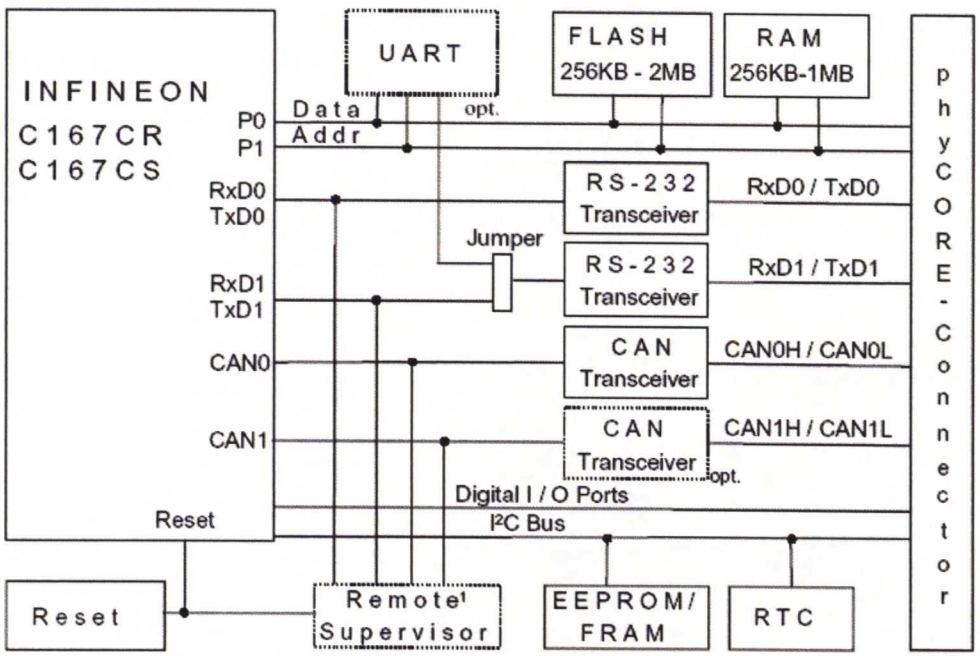


Figure 5-7 phyCORE 167 ***(Phytec Technologie Holding AG)***



¹: This feature is under development and is not available yet.

Figure 5-8 Block Diagram of the phyCORE167 processor board ***(Phytec Meßtechnik GmbH).***

The Communications, Camera and Decision Making Controller

The Mycable XXS1500 CPU module functions as a sensor and communications interface for the robot. It handles all image processing and communications tasks and passes relevant data and commands through its serial port to the Phytec phyCORE 167 CPU module.

The CPU in the XXS1500 module is an AMD/Alchemy (Au1500) MIPS 32 processor operating at 400MHz. It has 16 MB of flash memory, 64 MB of SDRAM, a Compact Flash card slot, two 10/100 megabit Ethernet connections, an USB 1.1 host, two serial ports and a PCI 2.2 /66Mhz initiator or target. The compact flash card slot houses the communications module (Bluetooth or WLAN) and the camera is connected to the USB host port. A block diagram of the processor board can be found below in Figure 5-9 *(Carstens-Behrens, M. 2004)*.

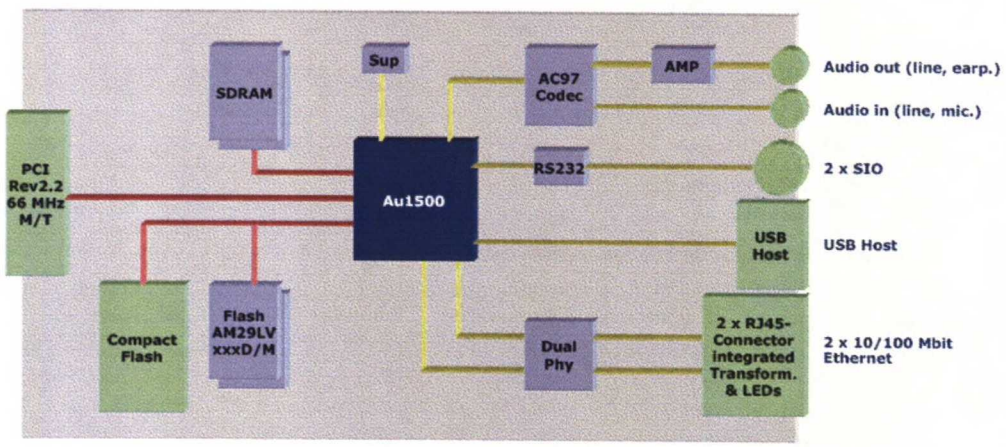


Figure 5-9 Mycable XXS1500 Block Diagram *(Carstens-Behrens, M. 2004)*

The backplane of XXS1500 has two step-down DC-DC power converters (from 8V to 14V down to 3.3V and 5V) for supplying the processor board with a single power supply. These are located on the backplane board of module. During testing one regulator chip supplying the 3.3V voltage to the processor and USB burned and was replaced. The absolute maximum current for the regulator chips is 2.5A. Care must be taken in connecting power to it so that the maximum current is limited. The backplane also has an external

3.3V and 5V power supply. On the backplane there is also a 25 – pin Sub-D connector which is not used in the design.

5.2.2 Sensors

The sensors used in the design are mostly used for motion control of the ball. The only sensors that are used to monitor relations to the outside world are the camera and the accelerometers (for detecting collisions). In a way the Bluetooth device can also be thought as a sensor as it can distinguish distances between itself and other Bluetooth devices. However as its primary usage is to handle communications, it is not detailed in this subchapter.

Accelerometers

The working principle of microaccelerometers differs significantly from standard accelerometers because of the very limited space available in micro devices. Where standard devices use springs, microaccelerometers use a minute silicon beam with an attached miniature mass. The structure supporting the mass acts as a spring while the air in the surrounding is used to produce a damping effect. A piezoresistor implanted on the beam or plate measures the displacement of the attached mass. The resultant voltage from the deformation of the piezoresistor can be correlated with the acceleration of the vibrating mass *(Hsu, T. R. 2002)*. In Figure 5-10 the working principle of a MEMS accelerometer is depicted.

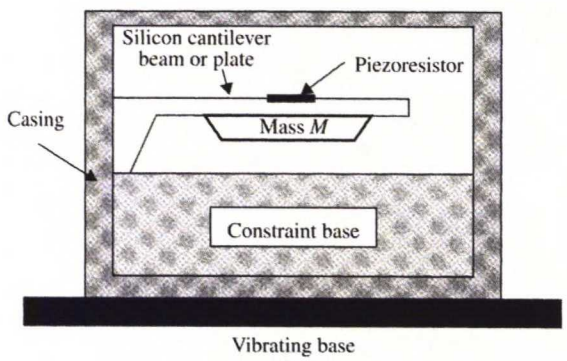


Figure 5-10 Working principle of a MEMS accelerometer *(Hsu, T. R. 2002)*

Three accelerometers / inclinometers of the SCA 610 series by VTI Hamlin are used for inclination and acceleration measurements. The purpose for the implementation of these sensors is to figure out the robots orientation in compared to the Earth's gravitational field and to function as a collision detection system. The accelerometers are situated each in a 90 degree angle compared to each other, on the x-, y- and z-axes respectively. They are relatively easy to use, requiring only the connection of a single capacitor between the ground (GND) and +5V (Vdd) pins. The inclination of the sensor is measured by connecting the output pin (Vout) of the sensor to the A/D converter of the microcontroller. Figure 5-11 shows the connection diagram of the accelerometer. The arrow in the picture indicates the positive direction for the acceleration.

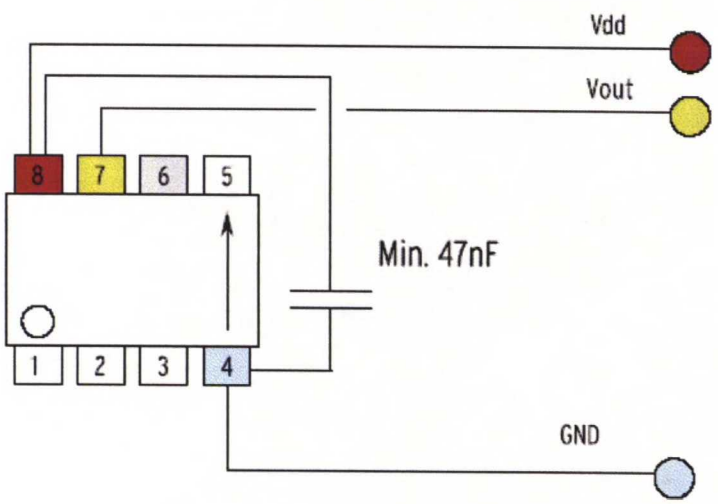


Figure 5-11 VTI Hamlin SCA610 connection diagram ***(VTI Technologies 2004)***

Gyroscope

A silicon ring gyroscope, CRS03-11, by Silicon Sensing Systems Japan is implemented in the design. It was chosen because of its high rate range of 573 °/s and good resolution bandwidth of 55Hz. The high rate range of the sensor is important as the robot has good turning ability. Usually the voltage output for an angular rate gyro will not be in the middle of the measurable range. The difference between the supposed zero angular velocity point and the real one is called the bias. The amount which the bias changes value during a certain time interval is called the drift of the gyro. The drift vs. time value for this

gyroscope is 0.55 °/s in any 30 sec period, which is quite normal for a piezoelectric gyroscope *(Silicon Sensing Systems Japan 2004)*.

The purpose for including a gyroscope in the design is to monitor the angular speed of the robot, to stabilize its driving and keep track of the current angle and the relation to the world coordinates. The bias of the gyro is measured by collecting gyro readings whenever the robot is stationary and calculating the mean from them.

Connecting the gyroscope is simple, with all the needed electronics already embedded on the gyro card. One just needs to connect three pins (Table 5-2 Pin Connections for the gyroscope), the operating voltage of five volts, ground and the rate output. The rate output is connected straight to the A / D converter of the microcontroller board.

The device is shown in Figure 5-12.

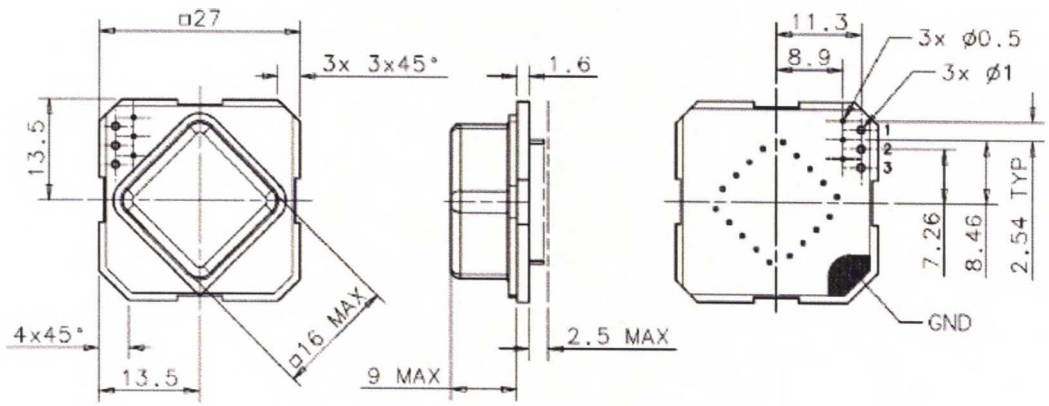


Figure 5-12 Silicon Sensing gyroscope model CRS03-11 ***(Silicon Sensing 2004)***

Table 5-2 Pin Connections for the gyroscope *(Silicon Sensing 2004)*

Pin Connections	
Pin	Function
1	+5V
2	Ground
3	Rate Output

Typical rate output for the gyro is depicted by formula (6):

$$V_o = \frac{1}{2} * V_{dd} + \left(R_a + SF + \frac{V_{dd}}{5} \right) \qquad [V] \qquad (6)$$

where SF is the scale factor. The value for SF (the calibration factor) is 3.49mV/°/s according the specifications for the device. Using the measured output V_o, the input voltage V_{dd} and the scale factor SF the applied rate R_a (°/s) can be calculated *(Silicon Sensing Systems Japan 2004)*.

Voltage Monitoring

The 12 V voltage of the energy system is monitored by the microcontroller. The battery voltage is measured by scaling it to fit the zero to five volt range measurable by the A/D converters. Scaling is done by division of voltage on the power management board. The schematic of this operation can be found in Figure 5-15.

Camera

A CCD (charge-coupled device) camera is connected to the XXS1500. It is used among other things for the recognition of objects and optical flow measurements.

5.2.3 Motors and Encoders

The motors used to drive the wheels are 2 W DC-motors from Maxon Motors. They have 16 – bit encoders connected to them. Without any gears the encoders count 16 pulses per rotation. With a gear size of 275, which the

motors have, and counting on both an up and down encoder pulse, the number of pulses counted per rotation of wheel is
16 * 275 * 2 = 8800.

To find the final number of encoder clicks in a single rotation of a wheel, we multiply this number by the wheel-to-motor gear ratio, which is three (to one). Thus the final value for encoder pulses per wheel rotation becomes
8800 * 3 = 26400.

A picture of the encoder and the pin allocation are shown below in Figure 5-13 and Figure 5-14 respectively.

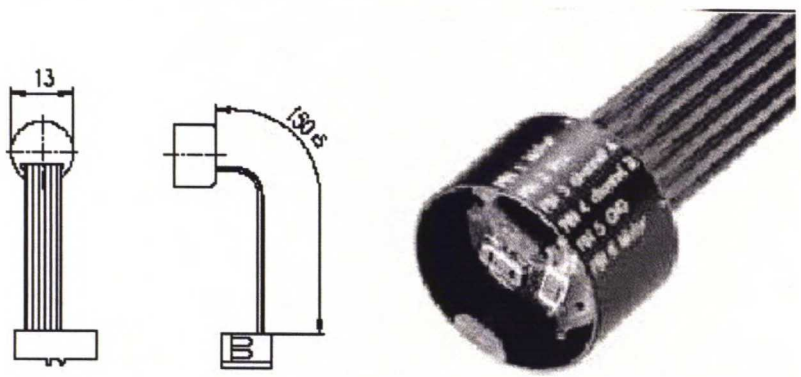


Figure 5-13 Maxon motor 13 mm digital magnetic encoder

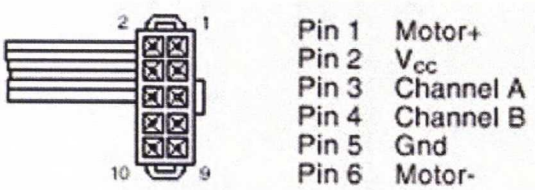


Figure 5-14 Pin allocation of the Maxon digital magnetic encoder

The pins one and six are used in running the motors, two and five supply power to the encoder (five Volts), while the encoder pulses come out of pins three and four.

5.2.4 Power Supply System

The basic requirements for the power supply system are handling enough energy, keeping all subsystems operational at the right times, and being able to

recharge the battery banks while the processors are operational. At first this problem was approached by having two separate sets of batteries of which only one was used at any given time to power the robot. This was thought to be a better solution than to make a parallel connection between the two banks using all the batteries for power at the same time and thus forcing the batteries to be reloaded while connected in the aforementioned configuration. However, tests showed rather mysterious drops in voltages when a load was connected to a battery bank. With the battery manufacturers giving internal resistance values of around 20 - 25 mOhms, this shouldn't have happened. Internal resistance measurements made in our laboratory showed that the values were closer to 250 – 400 mOhms, which explained the strange behavior. As it turns out, the values given by the manufacturers are measured with an alternating current power source connected to the battery itself (thus being impedance measurements), instead of measuring the internal resistance in a connection with direct current.

The large values of internal resistance in batteries led to the reconfiguration of the power electronics. A parallel connection between the battery banks, which was at first shunned, seemed like the best option to decrease internal resistance values. This helped all the subsystems to function at the expense of comfortable voltage measurement.

The robot draws energy from two battery banks connected in parallel each with ten AA-sized nickel metal hydride (NiMH) batteries connected in series. Each battery has a capacity of 2300 mAh. Care should be taken in handling the batteries and not letting any of the battery voltages drop too low. If this happens, or if the batteries heat up too much, the internal resistance of the batteries can rise to a high level. This can result in the batteries not being able to give enough energy to all the subsystems.

The voltage of the battery banks is measured by scaling the battery voltages to the range of zero to five volts, which is the input range of the microcontroller's analog-to-digital converter channels as well. At first, power to the robot was provided by one battery bank at a time. The selection of

which bank to use was handled by a relay controlled by the microcontroller using two of its digital output pins to change the relay's state. While one battery bank was recharging another's voltage could be measured accurately. With the two battery banks now connected in parallel the measured voltages will either be that of the banks' voltage while connected to the subsystems (thus lower than the voltage if there is no connection due to internal resistance) or the voltage with which the batteries are recharged.

In addition to the power provided by the batteries an additional auxiliary power source can be connected. The auxiliary power is designed to provide power when the robot is docked to a recharging station. When it is connected it provides power for the processors, while at the same time reloading the batteries. However the current it is able to put out to the batteries and to the Linux-box is limited to just over one ampere. This is because of the need to limit the battery reloading current to reasonable values (slow charge is 230 mA per bank, 460 mA in total).

One voltage regulator is used to make five Volts from the battery voltage or auxiliary power for the Phytec phyCORE 167. Power to the motors and the Linux-box is drawn straight from the batteries without any voltage regulation. At first, both the Linux-box and the motors were behind a ten-Volt regulator. This caused sudden drops in voltage when the motors were switched on or off, which in turn resulted in the Linux-box rebooting or even getting stuck when a sufficient voltage couldn't be provided to it.

The power supply system's electronics connection diagram can be seen below in Figure 5-15.

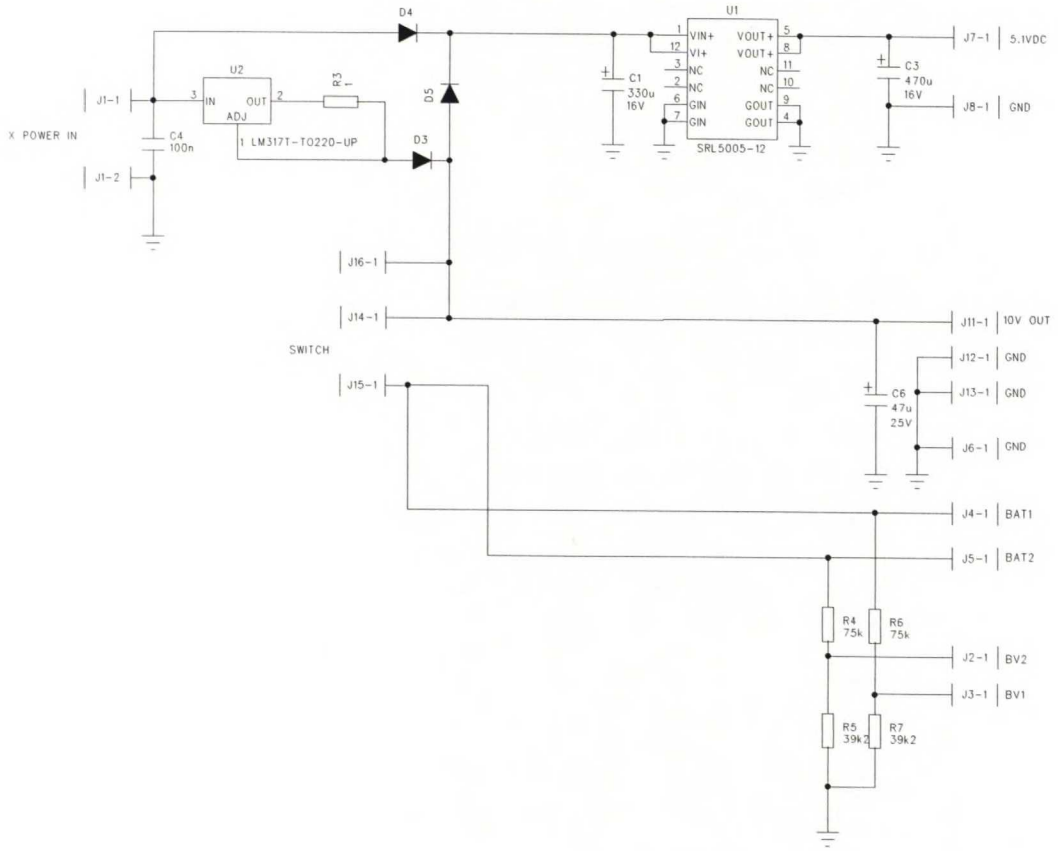


Figure 5-15 Energy electronics

5.3 Software

5.3.1 Tools Used

The programming of the Phytec phyCORE 167 microcontroller is done using the Keil Software µVision2 development environment. A real-time kernel, RTX-166, also by Keil Software, was used to handle the task management needed for the real-time control of the robot. The program itself was written in the C-programming language. The development environment features a version of an ANSI-C compiler. One of the biggest limitations faced in the programming process was the relative smallness of the different data types, in particular with the integer and long integer types, with the former being a 16-bit number and the latter a 32-bit one. In some calculations this resulted in

overflows of variables of the aforementioned data types and some serious debugging before the errors were found.

The real-time kernel provides the user with some nice tools to handle real-time operations. These tools are semaphores, which are used for mutual exclusion and sharing of resources, and mailboxes, which function as a kind of message buffer or a queue of messages for a task. If dynamic data structures are needed, separate memory pools have to be reserved from the memory available for the program.

The code for the controller has to be compiled to a binary format, namely HEX-86. The compiled file can then be loaded into the controllers Flash-memory by using Flashtools or a similar program while the controller itself is in the Bootstrap-loader mode.

5.3.2 Structure of the Code

The program written for the Phytec phyCORE 167 microcontroller board is depicted below in Figure 5-16 as well as the actuators the program is controlling or otherwise actively using. The notation used in the figure is that of a UML Component Diagram. It shows the dependencies of the different parts of the code, for example the motor control task (MC_task) is dependent on the positional information given by the location task (Locator _task).

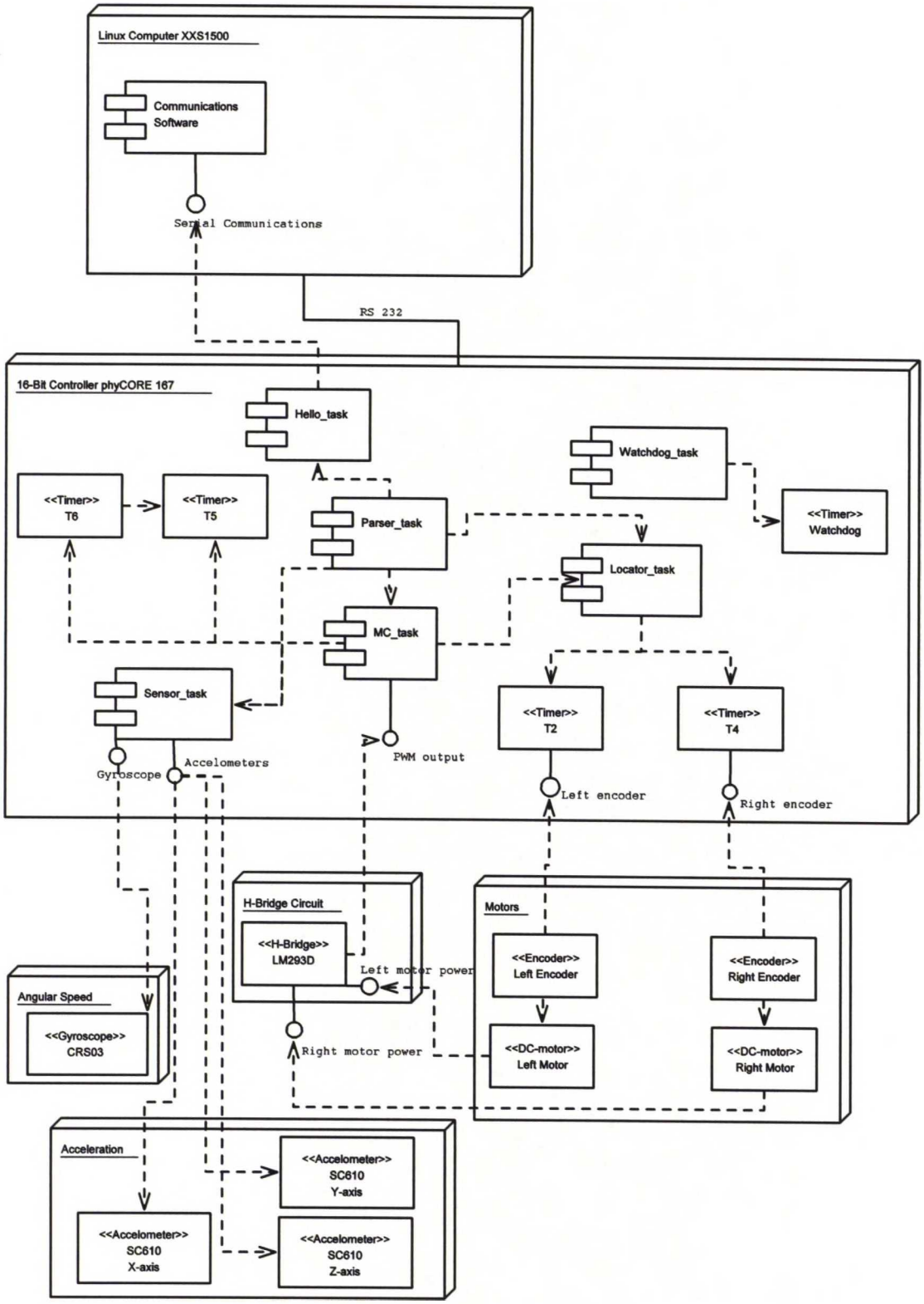


Figure 5-16 Structure diagram of Minirollo

Overview

The code consists of tasks and static libraries. The primary tasks are the main task, the locator task, the motor controller task, the communications

(Hello_Task) task, its extension which parses the given commands, the task which gathers sensor data (Sensor_task) and the watchdog task.

Start-up

The main task handles the initialization of the real-time operating system, input and output ports as well as the start-up of the other tasks. After starting up the system and the tasks the main task will end itself.

Odometry

The locator task tries to keep track of the current position of the robot by means of calculating the position and angle values from the motor encoder's own values. Timers T2 (left motor) and T4 (right motor) function in incremental encoder mode thus keeping track of the current encoder readings of the robot. All timers are 16-bit, thus they get values from 0 to 65535. In case of a timer over- or underflow in the encoder counter (timer) an interrupt is generated and one is added or subtracted from a global long integer variable (32-bit) which can be then used to calculate the total distance traversed by one wheel.

Motor Control

The motor controller task does two types of control, speed control for the wheels and position control based on the data from the locator task. It is basically a finite state machine with the two different states being speed and position control. The position control part of the motor controller task uses the speed control functions to drive where it is instructed to go. For the speed control a proportional-integral (PI) feedback-control algorithm is implemented. For both of the wheels proportional control is used separately. An error term is calculated from their differences in speeds and added to an integral term. This term is then subtracted or added to each of the wheels speeds modifying them to try to match each others' *(Jones et al, 1999)*. Figure 5-17 depicts the workings of the algorithm.

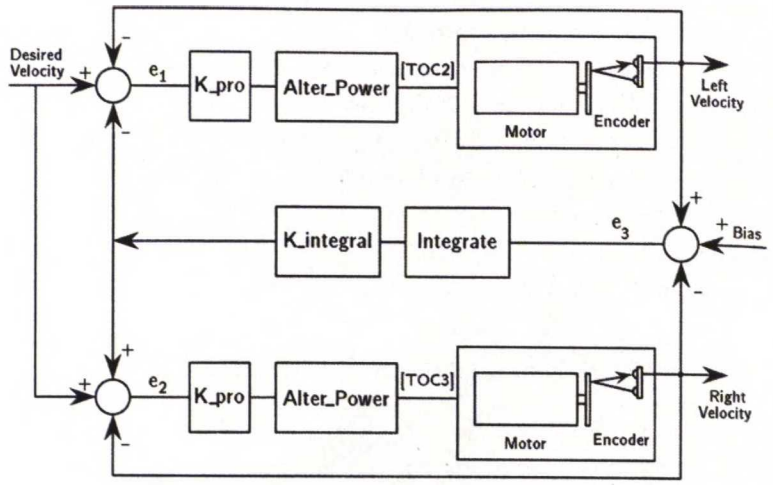


Figure 5-17 The coupled PI – speed control algorithm (Jones et al, 1999)

On the top and bottom are the proportional control loops, while in the middle there is the central feedback path for the integral controller.

Counters / timers T5 and T6 of the C167 processor are joined together, so that when timer T6 overflows it will add one bit to the counter T5. As counter T5 overflows it will add one to an unsigned integer (a 16 - bit number). Together T5 and T6 form a 33 bit counter (one bit from the interrupt flag) and the 16-bit variable extends it to 49 bits with a timer resolution of 12.8 us. The exchange of data between the motor controller and locator task modules is handled using common global variables and semaphores which handle access to these variables. The motor controller task queries positional values from the locator task and uses them for speed calculation in conjunction with the stored old positional values.

The motor control signals, done with pulse width modulation (PWM) are used in determining the speed of each wheel. Both of the wheels can be driven forwards and backwards. The microcontroller's PWM register values correspond to values between 0 and 16384. While a wheel is driven forwards, the PWM value 16384 denotes the maximum speed and the value 0 is the minimum speed value. While driving the wheel backwards 0 corresponds to maximum speed and 16384 to minimum. If the rotational direction of a wheel is changed without first shutting down the motors (preventing voltage from

coming to the H-bridge motor control circuit), a drop in voltages will result which will bring down the Linux box. The shutting down of the motors is accomplished by disabling the H-bridge enable bits.

Communications

The communications task uses the command parser task to interface with the motor controller task and issues commands to it. It also handles communications through the serial port with the XXS1500 by both receiving commands and sending data through it. Inbound communications should start with the dollar sign ($, ASCII 036) and end with the hash mark (#, ASCII 035). The commands listed in appendix B should be used to command the robot.

Data Gathering

The sensor task gathers data and periodically sends it to the Linux box.

Watchdog

The watchdog task is used to periodically service the watchdog timer. If the watchdog timer is not serviced during the set time interval and it overflows (all counter bits are ones and one bit is added to it) the controller will reboot. The watchdog is primarily used to avoid software lock-ups.

6. Testing the Odometry of the Robot

This chapter details the tests that were performed on Minirollo and their results. Their purpose was mainly to find out data of how the odometry was working. The odometry data was gathered solely from the motor encoders. This data could later be used to better calibrate the robot and to improve its control. Three tests were made, the first one to check the consistency of the distance measurement while driving straight, the second and third one to find out how good the calculation of the angular orientation.

6.1 Driving Forward

In this test, the robot was given an order to drive both motors at the same velocity (13 cm/s) for a set period of time (15 s). The robot would drift a bit to the left during the test, thus resulting in sideways motion. This sideways distance could be seen from the encoder measurements and is taken into account in calculating the total distance traveled. Only five test runs were made as the robot showed remarkable consistency in the distance traveled. This distance was verified by hand measurements (error in the region of 1 cm). The results are listed below in Table 6-1.

Table 6-1 Results from the forward driving test (in centimeters)

Distance from Encoders	Measured Distance
194,4604	193
194,4604	194
195,5911	194
194,1008	193
194,1739	194,5

The error mean from these results is 0,8146 cm and the standard deviation is 0,8254 cm. The calculated standard deviation is actually below the error of the

hand measurements, which shows that the internal distance measurements of the robot are quite accurate.

This and other tests that have been performed showed a significant difference between the frictions of each wheel, namely turning the left wheel requires more force. It also seems that at some point during that wheel's rotation the friction goes up and more power has to be applied to it to keep it rotating at a set velocity. This in turn results in occasional wobbles in the trajectory of the robot.

6.2 Driving Back and Forth

The second of the tests performed on the odometry of the robot involved giving it orders to drive forward for fifteen seconds at a velocity of 13 cm/s (this corresponds to the test detailed in the previous subchapter), turn 180 degrees and to drive back. The amount by which the robot deviated sideways from its starting spot after the total driven distance of 390 cm and the one 180 degree turn was measured. The results can be found below in Table 6-2.

Table 6-2 Measured sideways deviation (in centimeters)

Sideways Deviation from the Starting Point
-6,5
2,0
2,0
1,0
12,0
21,0
-2
11
0,5
16,0

The calculated variance from these measurements is 77 cm^2 and standard deviation I 8,8 cm. The average angular deviation is 1,3 degrees. The value for standard deviation of the angle does not seem to be very big, but it can result in a large positional error when longer distances are traversed. It also could be

much worse considering the shape of the robot, and the wheel slippage that is very much a reality for the low-friction wheels.

Currently, when the robot is accelerating or decelerating, the control of the wheels is not perfect. The differences in friction between the wheels cause them to rotate with error in their angular velocities, which usually results in the robot turning during steady acceleration. This can be fixed by tweaking the speed control parameters and perhaps even the algorithms.

6.3 Turning in Place

In the third and final test, the robot was ordered to rotate a full circle or 360 degrees. After the robot came to a halt the angle difference to the starting orientation was measured by three different means:

- Odometry data from the wheel encoders
- Data from differences in pictures taken with the robot's camera before and after the rotation (error +/- 1 cm)
- Pictures taken from above the robot with a handheld digital camera before and after the rotation, and angle measured by hand from the pictures (error +/- 5 cm)

The measured angular deviation values form the set zero value are shown below in Table 6-3.

Table 6-3 Angle measurements from different sources

Odometry	Robot Camera	Handheld Camera
2,5256	-4,296	-6
5,985	23,279	22
1,6102	8,021	4
5,5102	-2,897	-2
1,2033	-20	-25
-1,1707	-10,739	-12
0,4911	16,997	15
6,5615	2,697	0

The values from the handheld camera pictures are used to check if there are errors in the camera calculation. By comparing these values it can be seen that

they are very close to each other and within the error boundaries. The data from the robot's camera seems to be good.

Next statistical variables from the data sets collected from the odometry and the robot's camera are calculated. The standard deviation from the error of these two is calculated to be 13,49 degrees with a mean of 1,21 degrees. It seems that by solely using odometry measurements the angle estimation error will increase rapidly.

7. Conclusions and Possible Future Work

In this thesis the design of a member of heterogeneous robot society has been presented. The society it or its successors will eventually be a part of aims to fulfill a number of research goals, including the long term autonomy of the system and the testing of novel control architectures.

The robot society for which the robot is being developed requires a robot potentially capable completing a multitude things, but does not set tight constraints or definitions for how the robot should function. Flexibility is thus a must, as all the scenarios the robot will likely be performing are not fully defined yet.

One possible operating scenario is presented in chapter 2.4. 'The Night Watchman' is a scenario where a robot society can be used to perform interior surveillance in a building, physically track and try to identify possible intruders while they are present in the guarded area. The flexibility of moving robots which can communicate with each other and with the possible operator will make this scenario a viable alternative to old static camera based systems used in industrial and corporate security alike.

The robot which is described in this thesis is the first piece of a novel robot society. It has the ability to sense its environment, to communicate with other robots or an operator and it can move around quite well. As seen from the test results, if it has only the motor encoder data to navigate the angle estimate will degrade quite rapidly. With the implementation of the other on-board sensors, mainly the gyro, camera and accelerometers the angular orientation estimate can significantly be enhanced. The microcontroller and the electronics built around it are more than sufficient to control the basic operations of the robot.

External recharging of the batteries can be handled without affecting the operation of the processors. However, as no recharging system has been

designed yet, no external plugs or metal surfaces where to apply a potential difference have been implemented yet. Now the wires must be manually connected while the ball casing is removed.

It should be considered if the ball shape is necessary for the operation of the robot in the robot society. This shape poses some limitations to its operation as well as makes controlling the ball more difficult. With some simplification in the internal mechanics, and with the replacement of the halves of the ball with actual wheels, the ability to drive backwards would be gained. There would also be more room for sensors to see out, as now they must point out from the 2,5 cm gap when the robot is opened. The loss of the ball shape could make manipulating the robot more difficult by external grapplers etc. Also a protective shell that the ball casing offers would be lost, but it could be easily replaced another protective casing of some sort.

The Bluetooth technology provides a promising communications medium for short-range dynamic networks. By utilizing Bluetooth in the robot society communications between the robots should reliable with only few collisions and errors due to its robust frequency hopping properties.

The next challenge after getting Minirollo fully operational will be the building of the rest of the planned robot society. How the next prototypes will look and function like it has not yet been decided. The next, and the far greater challenge involves building a society capable of a multitude of tasks and long term autonomy.

8. References

Appelqvist, P. 2000. *Mechatronics Design of a Underwater Society – A Case Study of Minimalist Underwater Robots for Distributed Perception and Task Execution*, Ph.D thesis, The Helsinki University of Technology, Pica-set Oy, Helsinki, pp. 29-37.

Arkin R.C., Collins T.R.and Endo 1999, *Tactical mobile robot mission specification and execution*, Mobile Robots, (September 1999).

Asama, H., Matsumoto A., and Ishida Y. 1989, *Design of an autonomous and distributed robot system: ACTRESS*. In Proceedings of IEEE/RSJ International Workshop on Intelligent Robots and Systems, pp. 283-290, Tsukuba, Japan, 1989.

Asama, H., 1994. *Trends of Distributed Autonomous Robotic Systems,* Distributed Autonomous Robotic Systems, Asama, H., Fukuda, T., Arai, T., and Endo, I. (eds.), Springer-Verlag, Tokyo, pp. 3-8.

Balch, T. 2002. *Measuring Robot Group Diversity.* From *Robot Teams: From Diversity to Polymorphism*, Balch, T. and Parker, L. E. (eds.), A K Peters Ltd. Natick, Massachusetts, pp. 93-135.

Balch, T. and Arkin, R. C. 1994, *Communication in Reactive Multiagent Systems*, Autonomous Robots Vol. 1, pp. 1-25, 1994.

Barriere, L., Fraigniaud, P., Narayanan, L., Opatrny, J., *Dynamic Construction of Scatternets of Fixed Degree and Low Diameter* (Online), 2003. Available from: http://citeseer.ist.psu.edu/607056.html

Basagni, S. and Petrioli, C. 2002, *A Scatternet Formation Protocol for Ad-hoc Networks of Bluetooth Devices,* Dept. of Electrical and Computer Engineering, Northeastern University (2002) (Online), referenced 29.9.2004. Available from: http://www.tik.ee.ethz.ch/~beutel/projects/topologies/basagni_scatternets.pdf

Bluetooth SIG Specification of the Bluetooth System, Version 1.2—Core, November 2003 (Online). Available from:

https://www.bluetooth.org/foundry/adopters/document/Bluetooth_Core_Specification_v1.2

Bruno, R., Conti M., Gregori, E. 2001. *WLAN Technologies for Mobile ad hoc Networks* (Online). In: 2001 Proceedings of the 34th Hawaii International Conference on System Sciences, Maui, Hawaii, USA, 3-6 Jan 2001. Referenced 5.10.2004. Available from http://csdl.computer.org/comp/proceedings/hicss/2001/0981/09/09819003abs.htm

Carstens-Behrens, M., *Developer's Manual for the XXS1500 CPU Module* (Online). Mycable GbMH, September 2003, Aukrug Germany. Referenced 4.2.2005. Available from http://www.mycable.de

Dudek, G., Jenkin, M. and Milios, E, 2002. *A Taxonomy of Multirobot Systems.* From *Robot Teams: From Diversity to Polymorphism*, Balch, T. and Parker, L. E. (eds.), A K Peters Ltd. Natick, Massachusetts, pp. 3-22.

Dudek, G., Jenkin, M., Milios, E., and Wilkes, D. 1993. *A Taxonomy for Swarm Robots*, Proceedings of the 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), July 26-30, 1993, Yokohoma, Japan, pp. 441-447.

Dudek, G., Jenkin, M., Milios, E., and Wilkes, D. 1996. *A Taxonomy for Multi-Agent Robotics*, Autonomous Robots Vol. 3, pp. 375-397,, 1996.

Frodigh, M., Johansson P., Larsson P., *Wireless ad-hoc networking—The art of networking without a network,* Ericsson Review no. 4 ,2002 (Online), referenced 4.10.2004. Available from: http://www.ericsson.com/about/publications/review/2000_04/files/2000046.pdf

Grabowski, R., Navarro-Serment, L. E., Paredis, C. J. J., Khosla, P. K. 1999. *Heterogeneous Teams of Modular Robots for Mapping and Exploration*, Autonomous Robots - Special Issue on Heterogeneous Multirobot Systems.

Halme, A., Jakubik, P., Schönberg, T., Vainio, M. 1993. *The Concept of Robot Society and Its Utilization*, Proceedings of the 1993 IEEE/Tsukuba International Workshop on Advanced Robotics, November 8-9, 1993, Tsukuba, Japan, pp. 29-35.

Hsu, T. R., *MEMS & Microsystems Design and Manufacture*, McGraw-Hill, New York, 2002, pp. 62-65.

Johansson, P., Ericsson J.S., *Ad-hoc IP Networks over Bluetooth* (Online). Referenced 4.10.2004. Available from: http://www.isoc.org/isoc/conferences/inet/01/CD_proceedings/T59/INET_Bluetooth2col.htm

Jones, J. L., Flynn, A. L., Seiger, B. 1999. *Mobile Robots, Inspiration to Implementation*, A K Peters Ltd. Natick, Massachusetts, pp. 249-264.

Macker, J., Corson, S., Fenner, B., Zinin, A., *Mobile Ad-hoc Networks (MANET),* 2004 (Online), referenced 4.10.2004. Available from: http://www.ietf.org/html.charters/manet-charter.html

Mataric, M. J. 1992. *Minimizing Complexity in Controlling a Mobile Robot Population.* From Proceedings of the 1992 IEEE International Conference on Robotics and Automation, May 1992, Nice, France.

Mc Daid, C., *Bluetooth Tutorial*, Palo Wireless, Bluetooth Resource Center (Online). Referenced 4.10.2004. Available from http://www.palowireless.com/infotooth/tutorial.asp

McLurkin, J. 1995. *The Ants: A Community of Microrobots*, B.S. thesis, The Massachusetts Institute of Technology, Cambridge, Massachusetts.

McLurkin, J. 2004. *Stupid Robot Tricks: A Behavior-Based Distributed Algorithm Library for Programming Swarms of Robots*, M.S. thesis, The Massachusetts Institute of Technology, Cambridge, Massachusetts.

Melodia, T., Cuomo, F., *Ad-hoc networking with Bluetooth: key metrics and distributed protocols for scatternet formation* (Online). INFOCOM Department, University of Rome ''La Sapienza'', September 2003, referenced 29.9.2004. Available from: http://www.tik.ee.ethz.ch/~beutel/projects/topologies/melodia_adhoc2003.pdf

Mikéska, Z. *Radio Specification of the Bluetooth System* (Online), 2004, Institute of Radio Electronics, Faculty of Electrical Engineering and Communication, Brno University of Technology. Referenced 4.10.2004. Available from http://www.elektrorevue.cz/clanky/04003/english.htm

Mišić, J., Mišić,V., Chan, Ka Lok, *Talk and Let Talk: Performance of Bluetooth Piconets with synchronous traffic,* 2003 (Corrected Proof). Ad

Hoc Networks, Online journal, Elsevier Science. Department of Computer Science, University of Manitoba, Winnipeg, Manitoba, Canada R3P 0E2 and Department of Computer Science, Hong Kong University of Science and Technology, Hong Kong, PR China. Referenced 7.10.2004. Available from: http://www.sciencedirect.com/science/journal/15708705

Navarro-Serment, L. E., Grabowski, R., Paredis, C. J. J., Khosla, P. K. 2002. Millibots, IEEE Robotics & Automation Magazine, December, 2002, pp. 31 - 40.

Phytec Meßtechnik GmbH, *phyCORE-167CR / phyCORE-167CS Hardware Manual,* Phytec Technologie Holding AG, Robert-Koch-Str. 39, D55129 Mainz, Germany. April 2003.

Phytec Technolgie Holding AG, *phyCORE-167CR High Performance 16-bit with on-chip CAN (Online),* referenced 8.12.2004, available at www.phytec.co.uk

Silicon Sensing Systems Japan, *CRS03 (unpackaged) Angular Rate Sensor* (Online), 2003, referenced 3.2.2005. Available from: http://www.siliconsensing.com

The BlueZ Project, *BlueZ The Official Linux Bluetooth Stack* (Online), 2003, referenced 4.10.2004. Available from: http://www.bluez.org

Vainio, M. 1999. Intelligence Through Interactions – Underwater Robot Society for Distributed Operations in Closed Aquatic Environment, Ph.D thesis, The Helsinki University of Technology, Pica-set Oy, Helsinki, pp. 29-37.

Appendix A Circuit Boards

In this appendix pictures of the pin layouts of the circuit boards are presented. Figures A-1 to A-5 show the schematics of the mounting board for phyCORE 167 and its pin out. Explanations for the different pins' functionality can be found in the C167 processor manual by Infineon. The figures A-6 to A-9 show the main board and it schematics. The figures A-10 and A-11 show the overview of the power supply board, which is de

scribed in more detail in chapter 5.2.4.

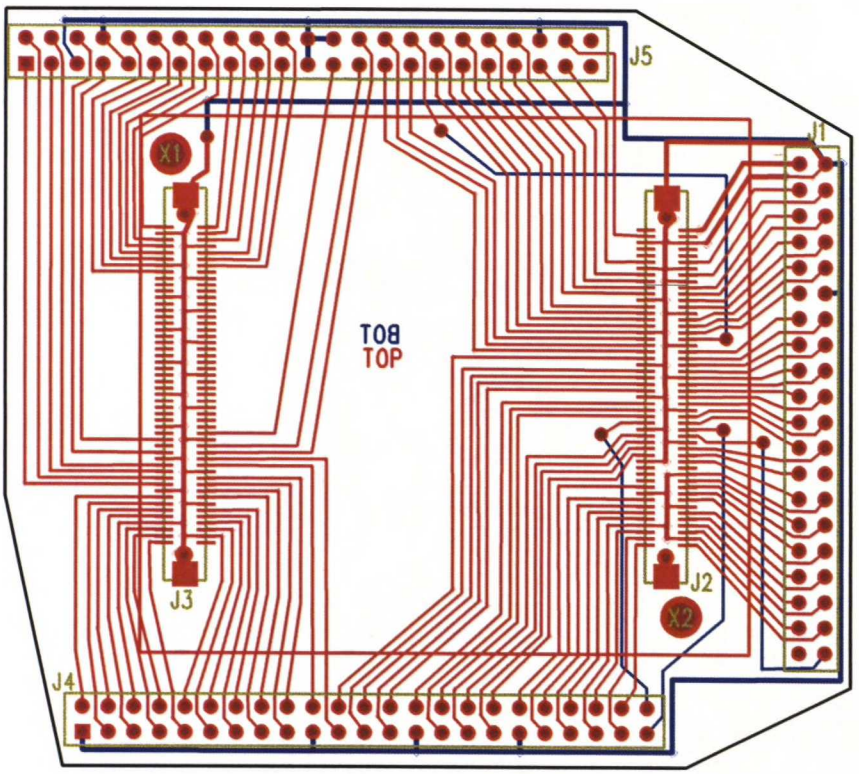


Figure A-1 Mounting board for phyCORE 167

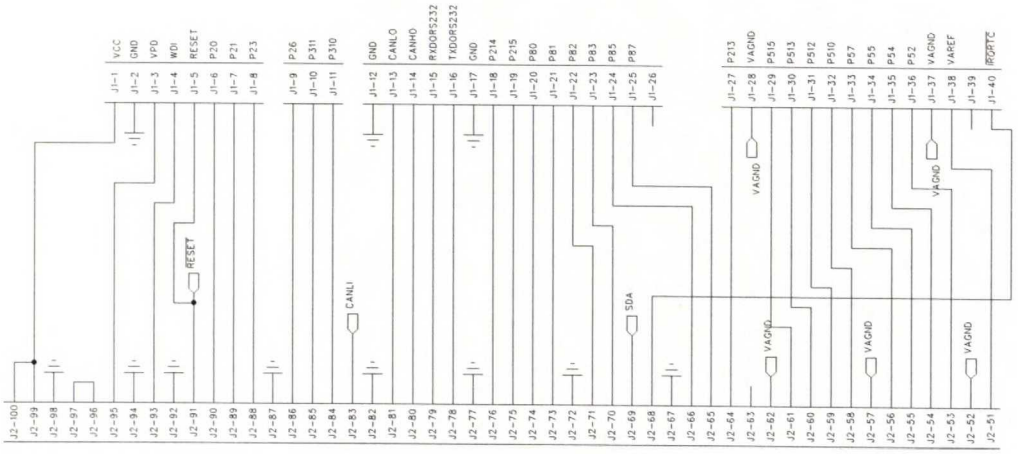


Figure A-2 Part of the phyCORE mounting board pin out

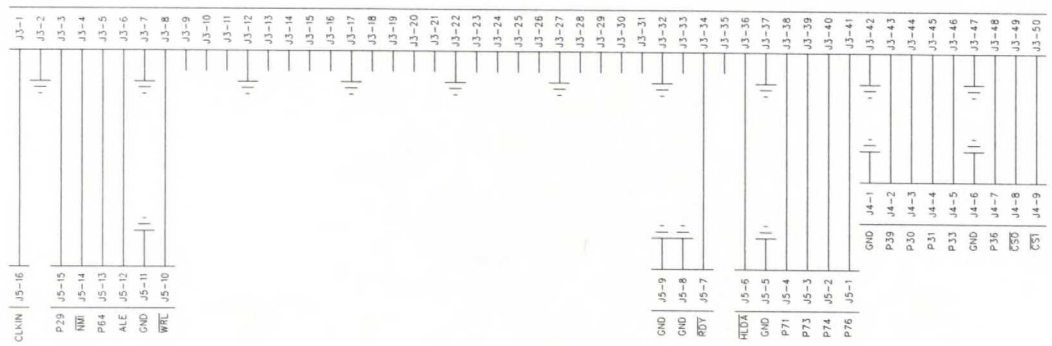


Figure A-3 Part of the phyCORE mounting board pin out

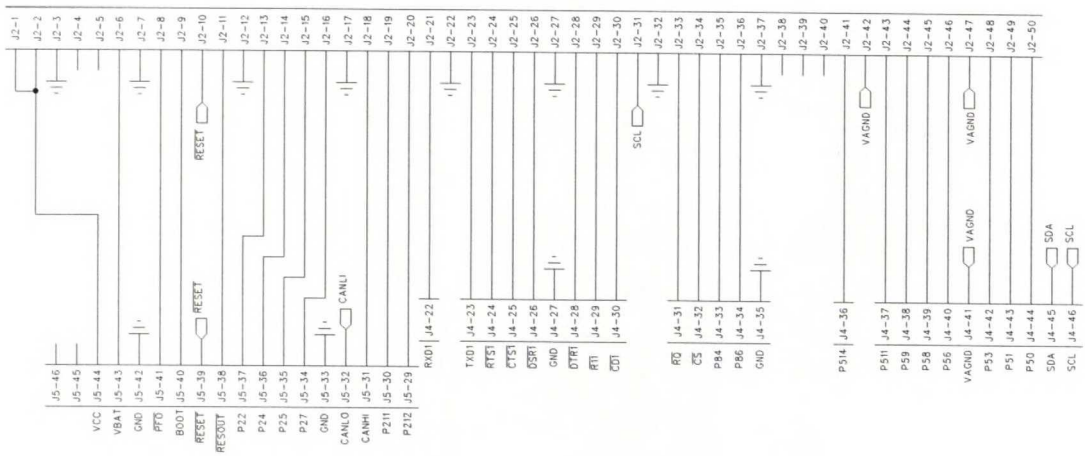


Figure A-4 Part of the phyCORE mounting board pin out

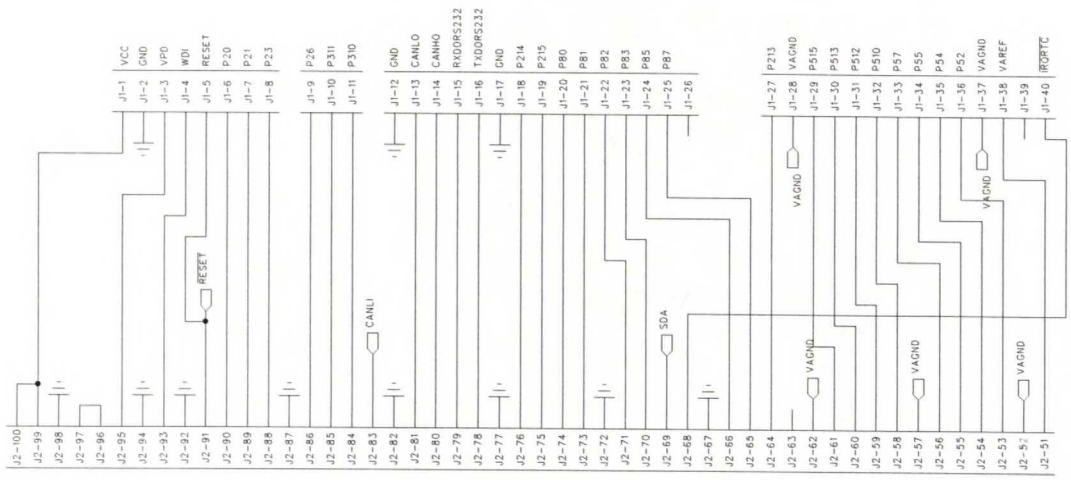


Figure A-5 Part of the phyCORE mounting board pin out

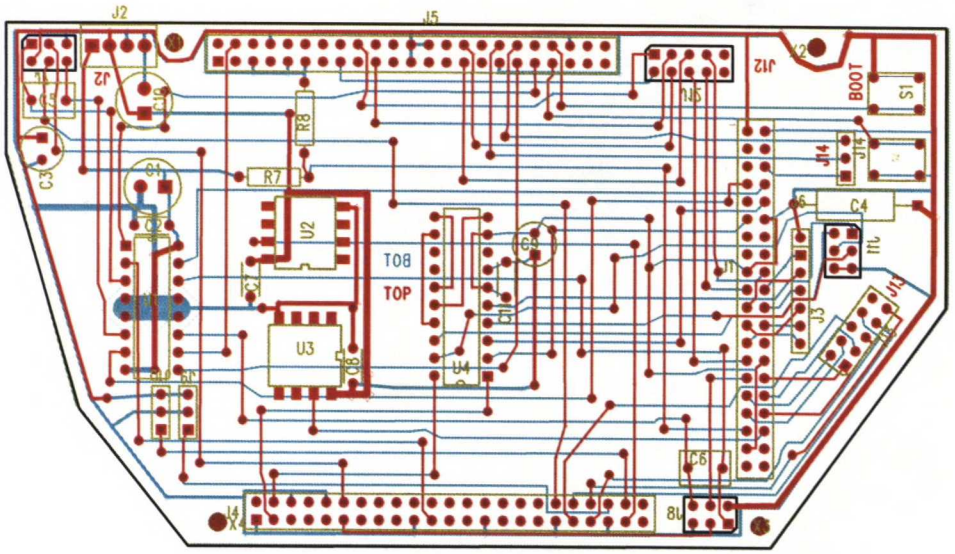
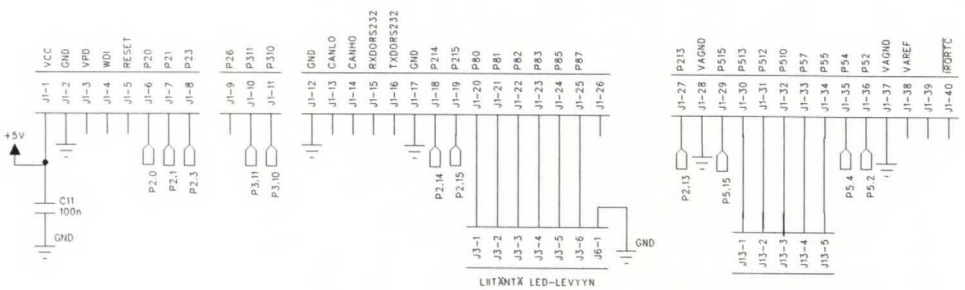


Figure A-6 The main board of the robot



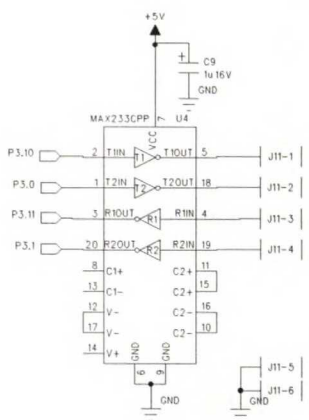


Figure A-7 Main board schematics

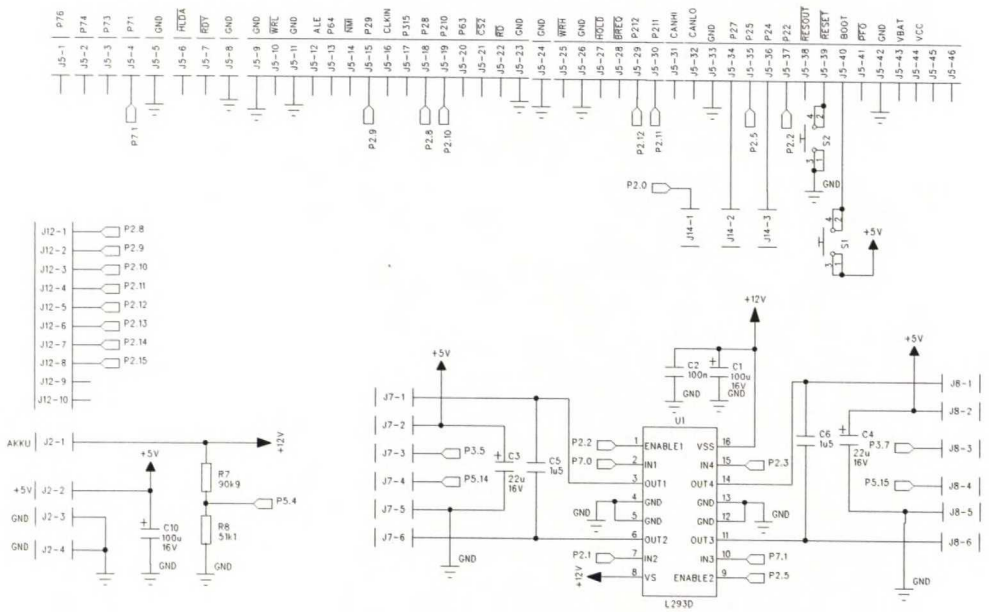


Figure A-8 Main board schematics

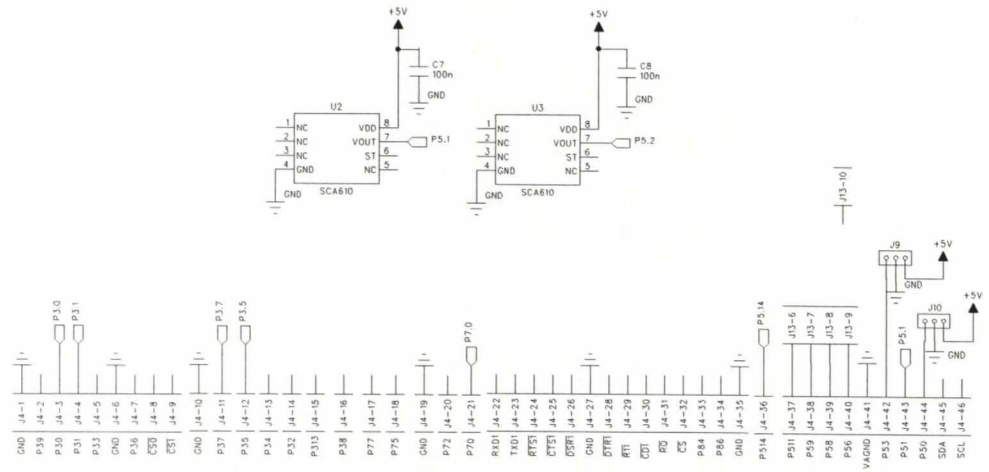


Figure A-9 Main board schematics

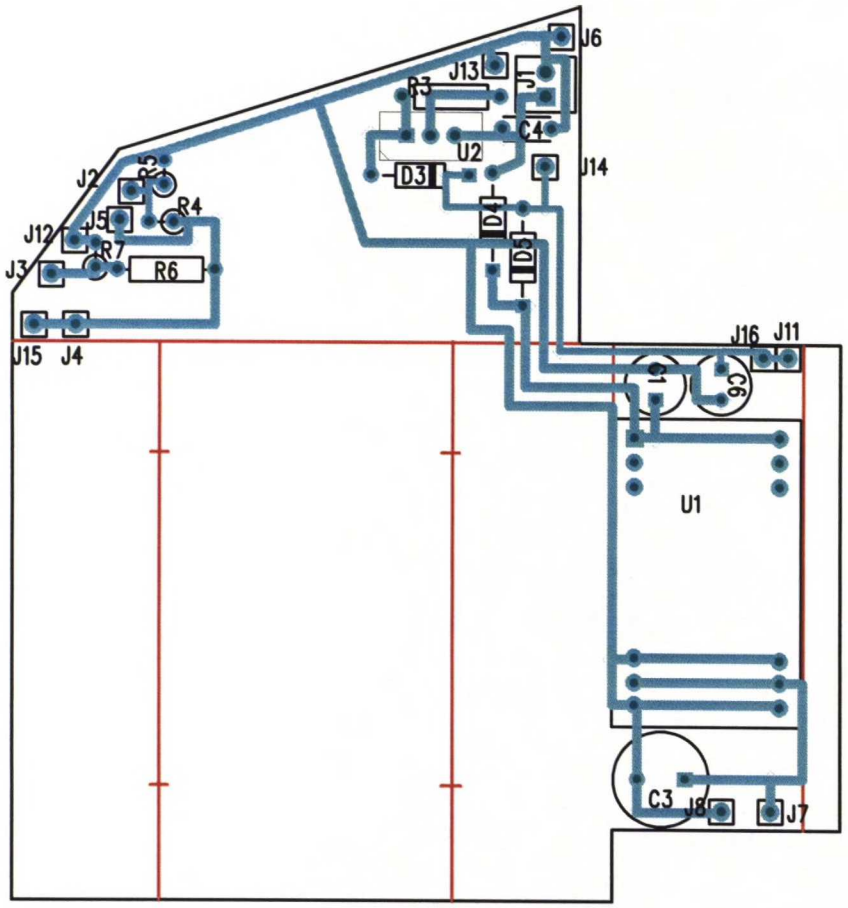


Figure A-11 Power supply board

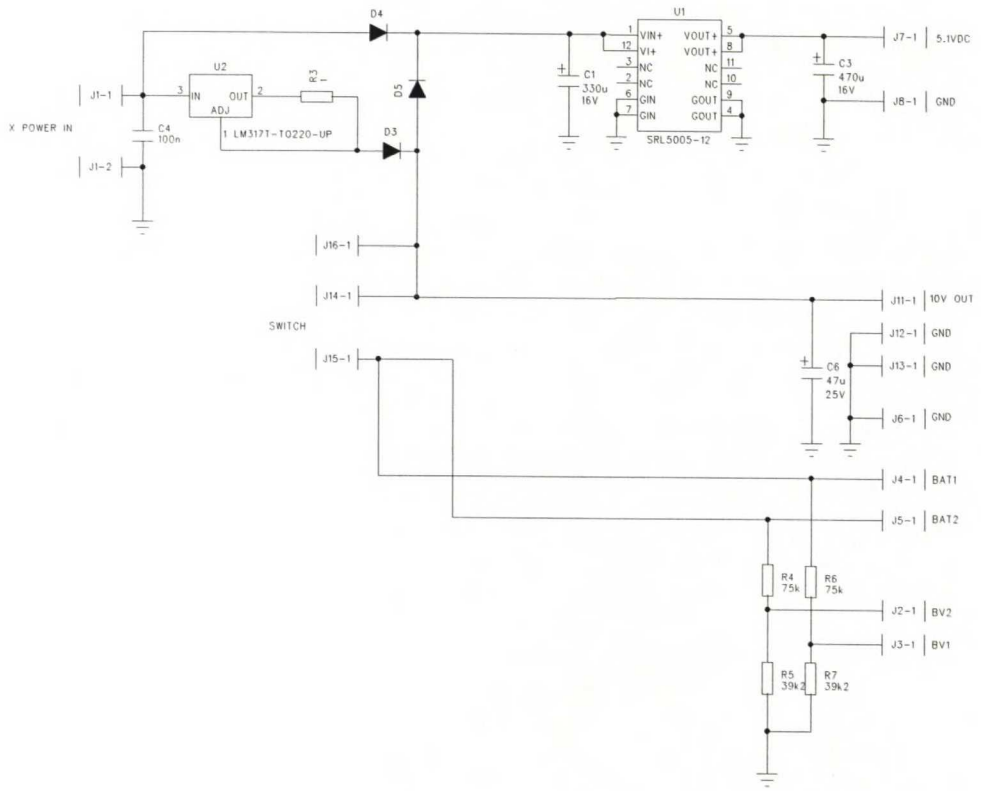


Figure A-10 Power supply board schematic

Appendix B Interprocessor Communications Protocol

The two processors used in the robot communicate with each other using the serial port (RS - 232). The messages are sent in ASCII – characters, with each message beginning with the dollar sign ($) and ending with the hashmark (#). The different commands which are currently implemented are listed below, but new commands are sure to be introduced after the writing of this. The commands are presented as they are to be inputted to each processor. Some of them require additional parameters, such as numerical values for denoting velocity, time etc. These are listed as denoted in ANSI – C string parsing and printing functions. The most commonly used are *%d* which signifies that an integer is to be inputted, *%f* which signifies a float and *%lu* which denotes a long unsigned integer. Some commands will have a return value; this will be given in the format described above.

B.1 Driving Commands

Arc Drive

INPUT: *$ARC %f %f#*

OUTPUT: none

Basic driving function for the robot. Speed values (in cm/s) need to be inputted.

Arc Drive for A Period of Time

INPUT: *$TIME ARC %f %f %d#*

OUTPUT: none

Drives the wheels by the given velocities (in cm/s) for a period of time (tens of milliseconds).

Close Robot Casing

INPUT: *$CLOSE#*

OUTPUT: none

Closes up the robot's casing.

Drive Forward

INPUT: *$FW %d #*

OUTPUT: none

Drives the robot forward for a given time period (given in tens of milliseconds).

Open Robot Casing

INPUT: *$OPEN#*

OUTPUT: none

Opens up the robot's casing.

Rotate

INPUT: *$ROTATE %f#*

OUTPUT: none

Rotates the robot by an angle given in the command (in degrees).

Rotate Left for a Given Time

INPUT: *$TURNL %d #*

OUTPUT: none

Rotates the robot left for a given time period (given in tens of milliseconds).

Rotate Right for a Given Time

INPUT: *$TURNR %d#*

OUTPUT: none

Rotates the robot right for a given time period (given in tens of milliseconds).

Rotate to World Angle

INPUT: *$ROTATE TO %f#*

OUTPUT: none

Rotates the robot to a world angle given in the command (in degrees).

Shutdown Motors

INPUT: *$SHUTDOWN#*

OUTPUT: none

Shuts down the motors by switching the H-bridge enable bits off.

Stop Motors

INPUT: *$STOP#*

OUTPUT: none

Stops down the motors by setting target speeds to zero.

B.2 Utility Commands

Get Location

INPUT: *$GET LOCATION# or $GP#*

OUTPUT: *$%f %f %f#*

Prints out the robot's current position and orientation (x, y, angle).

Print Collected Data

INPUT: *$DATA#*

OUTPUT: *$%lu %lu ...%lu#*

Prints out a 15 x 100 matrix of data collected from different registers.

[0] = Measurement time

[1] = Speed Control Mode

[2] = Left Encoder Value

[3] = Right Encoder Value

[4] = Gyro

[5] = Acc x

[6] = Acc y

[7] = Acc z

[8] = Left Encoder Extension

[9] = Right Encoder Extension

[10] = Left PWM Value

[11] = Right PWM Value

[12] = Robot Location on x-Axle

[13] = Robot Location on y-Axle

[14] = Robot Angle

Read Value from Analog to Digital Converter

INPUT: *$AD %d#*

OUTPUT: *$%d#*

This function returns a value from a specified A/D channel, with an integer value from 0 to 1024.

Channel Numbers:

1: Accelerometer x

2: Accelerometer y

5: Accelerometer z

7: Battery Voltage

10: Battery Voltage

11: Gyro

12: Ball Open Switch

13: Ball Closed Switch

Reboot Microcontroller

INPUT: *$REBOOT#*

OUTPUT: none

Reboots the microcontroller phyCORE C167, starts tasks and resets registers etc.

Set Location

INPUT: *$SET LOCATION %f %f %f #*

OUTPUT: none

Sets the robots positional values to those given in the command (x, y, angle).